공학석사학위논문

One-Class Classification by Norm Ball Covering

노름 공 덮개를 이용한 단일 클래스 분류기

2017 년 8 월

서울대학교 대학원

산업공학과

김 세 화

One-Class Classification by Norm Ball Covering

노름 공 덮개를 이용한 단일 클래스 분류기

지도교수 이 경 식

이 논문을 공학석사 학위논문으로 제출함

2017 년 6 월

서울대학교 대학원

산업공학과

김 세 화

김세화의 공학석사 학위논문을 인준함

2017 년 7 월

위 원 장 홍 성 필 (인)

부위원장 이 경 식 (인)

위 원 조 성 준 (인)

Abstract

One-Class Classification by Norm Ball Covering

Sehwa Kim
Department of Industrial Engineering
The Graduate School
Seoul National University

One-class classification (OCC) is a supervised learning technique for classification, where the classifier is constructed only by training the objects in the target class and determines whether new ones belong to the class or not. There have been attempts to solve OCC problem such as Support Vector Machines (SVM), Parzen Density Estimation (PDE) and other methods commonly used in multi-class classification.

In this paper, a new approach to OCC was proposed. The classifier consists of norm balls covering the objects in the target class: an object is classified in the target class if at least one of the norm balls covers it, otherwise it is rejected. We presented an algorithm of generating finite norm ball candidates. Then, by applying two conditions for 'good' norm ball the final candidates were chosen out of the candidates. An integer programming model for the selection

of the optimal norm balls was solved so that the norm balls with the minimum number effectively detect the target objects with the good predictive power.

The experiments were carried out to test the overall performance of our classifier using some artificial and real data from UCI Repository. The results showed that proposed model was comparable to OCC methods in the comparison group. Also, it had high sparsity leading to low testing burden compared to the other classifiers. In the noise experiment, maximum norm ball classifier was robust to noises.

Keywords: One-Class Classification, Norm Ball Covering, Integer Programming Model

Student Number: 2015-22855

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Backgrounds

Classification in machine learning is a fundamental step for analyzing the data. The classifier is constructed based on the given training data and predicts the class labels of unseen data. The traditional classification *i.e.* multi-class classifition focuses on how well it determines the appropriate class labels out of the pre-specified ones. For example, Support Vector Machines (SVMs) by Cortes *et. al.* [6] finds the hyperplanes dividing the space between the classes by solving a convex optimization problem.

However, One Class Classification (OCC) introduced by Moya *et. al.* [18] is to build a classifier which describes the target class and identifies only its objects. OCC seems similar to conventional binary classification problem which is the special case of multi-class classification for given two classes, but there are some differences between OCC and binary classification.

(a) OCC uses only target objects in the training, while binary classification needs both target and non-target ones. Thus, OCC methods are available when non-target class is poorly sampled or there are unlabeled

data, which is the mixture of target and non-target data.

(b) A one-class classifier can deal with the classification of unknown labeled data because it exactly captures the domain of the target class, but the binary classifier labels only target or non-target class. OCC can solve multi-class classification problem by constructing one-class classifiers for every class.

(c) When new non-target objects are observed and the classifier should update this information, it requires training process again. On the contrary, one-class classifier remains the same because it still rejects non-target objects.

Under the characteristics above, OCC is used in a broad range of application domains. For example, in machine monitoring there is large-sized normal data (target objects) obtained easily: on the other hands, the frequency of malfunction is very low, so it is hard and time-consuming to observe abnormal data (non-target objects). Then, one-class classifier trained by the normal one makes use of detecting abnormal ones *i.e.* outliers which deviates from normal data description. Other applications include tumor detection from magnetic response images (MRI) [36], masquerade detection [32] in a network, fraud detection for cards, insurances, text classification [35, 34] and others. It seems that a one-class classifier is very helpful to detect outliers, anomalies or novelties which represent unexpected pattern in the data set. In result, many techniques of OCC appear in Outlier detection [22] and Novelty Detection [2].

1.2 Literature Review

Many methods have been proposed to solve OCC problem. The basic idea for OCC is to understand the distribution of the target class. The straightforward approach to construction of a one-class classifier is to calculate the local density *i.e.* estimate the probability density function from the given training data containing only target objects: the classifier accepts an observation which local density is higher than or equal to a certain threshold, otherwise rejects it. The following is the standard methods based on local density estimation for solving OCC.

Mixture of Gaussians (MoG) [7], assumes that the probability density function (pdf) of the target class follows a linear combination of several Gaussian functions. This method produces a classifier with a smaller bias than one from single Gaussian density function. The number of Gaussian functions, the mean and covariance matrix for each Gaussian function are decided beforehand.

Parzen Density Estimation (PDE) [19] measures the local density in a non-parametric way without the assumption of a certain probability density function. For a test object $\mathbf{z}$, the method estimates pdf $p(\mathbf{z}) = \dfrac{K}{N/V_h}$ where K is the number of the points in the limited space R centered at $\mathbf{z}$, N is the total number of the data and V_h is the volume of R with width h (the length of window or bandwidth). K is the sum of identical indicator functions, each of which equals 1 if a training data point is inside R centered at $\mathbf{z}$, otherwise 0. This is the naïve version of PDE, which yields histogram-shaped pdf. For smoothing, various kinds of kernel function can be applied. That is why PDE is often called

Kernel Density Estimation (KDE).

Another local density based approach to OCC is to utilize nearest neighbors. Nearest Neighbor Method [27] for OCC is that the local density of a test object $\mathbf{z}$ is compared to that of its nearest neighbor based on the distance. The distances d_1 and d_2 are defined: d_1 is the distance between $\mathbf{z}$ and its nearest neighbor $NN(z)$ in the training data and d_2 is the one between $NN(\mathbf{z})$ and its nearest neighbor $NN(NN(\mathbf{z}))$. It is sufficient to compare these two distances because the local density of $\mathbf{z}$ and $NN(\mathbf{z})$ have the same fraction of 1/(total number of data points) individually but d_1 and d_2 related to volumes are different. If $d_1/d_2 <= 1$, then $\mathbf{z}$ is only accepted. It seems thatn their method is useful when there are low samples. Nearest Neighbor Methods considers the first nearest neighbor and can be extended to kth nearest neighbor's version.

There are techniques based on Support Vector Machines (SVMs). Schölkpf [23] assumed that in the feature space obtained by the use of a kernel function, there would be a hyperplane separated from the origin with the maximum margin. Instead, Tax & Duin [26, 28] attempted to use a hypersphere for data description. This method is called Support Vector Data Description (SVDD). In their theory, the algorithm finds a center and radius to minimize the volume of the hypersphere. The authors proved that SVDD is equivalent to Schölkpf's method. Both the two techniques solve quadratic optimization problem, minimize regularization and the training errors. Its optimal solution implies that the support vectors with positive dual values contributes to building the classifier. The use of a kernel function makes it possible for both to have more flexible data description of the target data.

There are other techniques for OCC even though they are not originally devised for OCC. The examples of decision trees (DT) for OCC can be found. [14] and [15] modified the existing DT algorithms for OCC problem and used both target and non-target data for training. In [12] and [17] neural network framework, which is called auto-encoder network, was applied in order to be trained only by target objects. Skabar [25] also presented a neural network algorithm where unlabeled data are always in need in the course of training process. In addition to the above methods, a variety of solutions to OCC have been under research.

1.3 Motivations

In this section, some advantages and disadvantages of previously mentioned OCC methods are presented. From them, we draw the motivations of this research and our new approach to OCC.

Though MoG requires relatively many parameters such as an average vector and a covariance matrix for each function, they are efficiently found by employing Expectation Minimization (EM) algorithm [3]. Once the parameters obtained from the algorithm are stored, then the testing process tends to be less tricky except the calculation of the inverse of the covariance matrix. MoG is less sensitive to noises rather than the use of single Gaussian functions, which is included in the training data and can result in the poorly estimated covariance matrix [29]. However, it could be very difficult for the users to intuitively decide the number of Gaussian functions.

Naïve PDE estimates only one parameter, bandwidth h. The training cost is almost zero while the classifier stores all the training points during the test phase. It gives us two implications: One is that if the samples in training data nicely represents the target class, then PDE is robust to noises. Otherwise, it needs sufficient size of samples to achieve the good performance. The other is that the sparsity [10] of the classifier is low, which causes testing burden and low interpretability in the original space when a kernel is used. This is because the model has non-zero terms as many as the number of training points. Hong *et. al.*[11] made efforts to obtain sparse PDE classifier. Given Gaussian kernels, the algorithm searches only meaningful terms and adjusts each bandwidth of

the selected kernels related to them in the light of minimization of integrated square errors.

In Nearest Neighbor Method, there is no user-defined parameters. It is nearly inexpensive for training wheares the calculation of the nearest neighbor corresponding to every test object is computationally expensive. Simultaneously, all the training points should be always stored as the references (or prototypes). The performance of this method is highly affected by noises by considering only one nearest neighbor. To overcome these drawbacks, Cabral *et. al.* [4] proposed an algorithm of reducing the reference points out of training data and considering more than the first nearest neighbors. Although this approach is based on the local density, it can be viewed as a instance-based method. In terms of interpretability, it just gives us the information of distances rather than more meaningful interpretation like the main components or factors of classification.

SVM based methods like SVDD have two types of parameters: one is associated to the trade-off between bias and variances and the other is kernel-related parameters. Especially, the former controls type II error, the rate of rejection of target data, so the classifier deals with noises when they are in the training data. It takes time to obtain the classifier during the training phase but there are many efficient algorithms. By solving the optimization problem, SVDD selects support vectors describing the data among training points, which leads to relatively high sparsity compared to PDE and Nearest Neighbor Method. Thus, it goes through less heavy testing process. However, interpretability could be difficult by the use of a kernel function.

DT has a strength for interpretability compared to other OCC algorithms

because it actually describes the data based on combination of intervals in the original space. But, it requires target data as well as unlabeled data or generation of artificial data outside target class for training. To overcome this drawback, Jeong & Choi [13] proposed two approaches of generating hyper-rectangles only by using training data. One is to merge intervals, and the other is to identify clusters in the training data and then generate hyper-rectangles.

To sum up, the properties for a good OCC technique can be summarized as follows.

(a) Interpretability in the original space: if the classifier nicely describes the target class, it is easy to extract simple classification rules or the important features from the model without certain tricks such as kernel functions. It helps to define the problem or understand the system in reality.

(b) Sparsity & Testing Burden: the model with high sparsity has the minimum number of non-zero terms, which leads to the reduction of storage for them. In result, it does not need to make a lot of efforts when testing new observations. High sparsity can be a significant issue when classification task is conducted within devices with limited computing and storage resources.

(c) Sensitivity to Noises: OCC uses only samples in the target class, so the structure of the classifier can be destructed by some noises in the training data. It should be guaranteed that OCC method shows the stable performance regardless of the existence of noises.

(d) The number of hyper-parameters: learning process can be expensive when OCC method allows a user to handle many hyper-parameters. It means that the number of them should be lower or at least they are encouraged to be intuitively estimated.

(e) Training/Learning process: the construction of a classifier should be efficient even if it has other good properties for classification task.

In this paper, we propose a novel approach to OCC which attempts to hold as many as the properties above. Our method introduces norm balls as a data descriptor and the concept of covering by them. For interpretability, hyper-rectangles would be a good choice because it finds it possible to comprehend the data by the disjunction of intervals. However, a norm ball also represent the distribution of the target class only with its center and radius. If maximum norm is applied, then the there is a similar effect on interpretation like hyper-rectangles.

Secondly, norm ball covering can be brought into positive intuition that if the norm balls properly covering the target objects are selected with minimum number, high sparsity is easily attained. Also, testing new observations is inexpensive because their labels are easily determined just by computing the distance from the centers of the minimum number of norm balls.

Thirdly, it can be observed more often that the target objects for the training phase not only has non-convex shape like separate clusters in the target class but also contains some noises. By covering the objects with norm balls which properly excludes the noises, it could expected that one-class classifier is robust

to noises.

Lastly, the use of norm balls possibly reduces the number of hyper-parameters related to the generation of them because they are characterized only by a center and radius. In contrast, it is cumbersome to find a lower and upper bound for an interval on every component if hyper-rectangles are chosen for a data descriptor. It gives the expectation that more hyper-parameters are needed compared to norm balls.

Our study presents an algorithm of generating polynomial number of norm ball candidates with the given target class data. It is also shown that with the information related to distances between the target objects, the parameters can be easily set up depending on the need of a user. For the given candidates, the norm balls which a norm ball classifier consists of are selected by solving an integer optimization problem to minimize the number of the selected norm balls for high sparsity. Then, experiments are conducted to examine how our classifier behaves in term of many aspects including sparsity, sensitivity to noises and others.

This paper is organized as follows. Section 2 shows the basic theory of norm ball classifier and how it is obtained. In Section 3, the performance of our classifier is evaluated and the properties mentioned before are explored through computational experiments. Section 4 includes discussions and the future research topics.

Chapter 2

Norm Ball Classifier

2.1 Definition of Norm Ball Classifier

In this section, We present the basic concept of Norm Ball Classifier (NBC) and how it works as a classifier. First, the definition of norm should be considered. *Norm*, more generally, *p-norm* of $\mathbf{x} \in \mathbb{R}^n$ is $\|\mathbf{x}\|_p = (\sum_{j=1}^{n} |x_j|^p)^{1/p}$ for $p \in \mathbb{N}$. If $p = \infty$, then $\|\mathbf{x}\|_\infty = \max_{j=1,\dots n} |x_i|$ also called maximum norm. Typically, norm is a function for measuring the length of a point in n-dimensional space.

For given $\mathbf{c} \in \mathbf{R}^n$ and $r \in \mathbf{R}_{++}$, we define a *norm ball* $B_r(\mathbf{c}) = \{\mathbf{x} \in \mathbf{R}^n : \|\mathbf{x} - \mathbf{c}\|_p \leq r\}$, which is a bounded and closed set. It is easily seen that a norm ball is specified by the two elements, center $\mathbf{c}$ and radius r. For example, 2-norm ball (maximum norm ball) is a circle (square) in $\mathbb{R}^2$ and expands to hypersphere (hypercube) in more than 2-dimensional space respectively. The actual shape of a norm ball in $\mathbb{R}^2$ depending on p is shown in Table 2.1.

Table 2.1: Examples of norm balls in $\mathbb{R}^2$

1-norm ball	2-norm ball	maximum norm ball

Let B a index set of finite norm balls. Now, suppose that there is a indication function I_{b_i} corresponding to norm ball b_i $i \in B$. For a input vector $\mathbf{z} \in \mathbb{R}^n$,

$$I_{b_i}(\mathbf{z}) = \begin{cases} 1 & \|\mathbf{z} - \mathbf{c}\|_p \leq r \\ 0 & \text{Otherwise} \end{cases}.$$

Norm ball classifier $f(\mathbf{z})$ is modelled as follows.

$$f(\mathbf{z}) = \vee_{i \in B} I_{b_i}(\mathbf{z})$$

This equation implies that if $f(\mathbf{z})$ equals 1, then the classifier accepts $\mathbf{z}$ and identifies it as a target object: otherwise, $f(\mathbf{z})$ becomes 0 and $\mathbf{z}$ is rejected. Figure 2.1 helps to completely understand how NBC behaves. In this figure, there are four 2-norm balls covering target data. According to the definition of NBC, $f(\mathbf{z}) = \vee_{i=1}^{4} I_{b_i}(\mathbf{z})$. Since $f(\mathbf{z}^1) = 1$, this classifier accepts $\mathbf{z}^1$. In contrast, it rejects $\mathbf{z}^2$, which belongs to none of the balls. Only 2-norm balls are used in the example but other p-norm balls can also participate in data description.

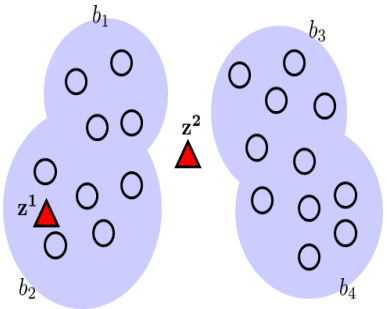


Figure 2.1: An example of NBC: the black circles stand for target data.

2.2 Construction of Norm Ball Classifier

In the previous section, It is explained how NBC classifies the observations. Then, the next question is to practically create the norm balls for NBC. Let $\{\mathbf{x}_l\}_{l\in N} \subseteq \mathbb{R}^n$ denote a set of training objects from the target class. In fact, an infinite number of norm balls can be candidates in $\mathbb{R}^n$. However, a finite set of norm balls is produced by employing only training data in our study. Then, the most appropriate norm balls are selected among the norm ball candidates by solving an integer programming model. Finally, the classifier consists of the selected norm balls from the model.

2.2.1 Generation of Norm Ball Candidates

For generation of norm ball candidates, there are two points. One is the fact that the target data for training is only available. The other is to focus on center and radius candidates capable of generating norm balls. In this situation, the best choice is to regard all the training points as center candidates. Radius candidates can be discretized: the p-norm distances between a center candidate (one of the training points) and the other points are only considered. Each center candidate has at most $|N|-1$ radius candidates. Consequently, the total number of the norm ball candidates is $O(|N|^2)$.

However, all of the generated norm balls do not properly contribute to the description of the target class. For instance, a norm ball with a center most distant from the inner of the target class distribution and its maximum radius have a lot of sparse space because many points in the ball are on one side. In

order to choose the final candidates as *good* data descriptors, they are forced to meet the two conditions.

(a) The first condition limits the size of radius r of a norm ball candidate: $\underline{\gamma} \leq r \leq \bar{\gamma}$ where $\underline{\gamma}$ and $\bar{\gamma}$ is the lower bound and upper bound of r respectively. If r is too small, it leads to overfitting by covering the center only. When r is highly large, the norm ball with the radius always accepts points regardless of their labels. This means that the classifier loses the prediction power.

(b) The second condition is that the center of mass of data in a norm ball $\mathbf{c}_M$ should be close enough to the center of the ball $\mathbf{c}$: $\|\mathbf{c} - \mathbf{c}_M\|_p \leq \mu$ where μ is a positive number. Although the radius of a norm ball satisfies the first condition, the points in the norm ball could shifted to one side in the ball. In this case, the norm ball with smaller radius containing the points is preferred because it has higher density than the previous norm ball.

$\underline{\gamma}, \bar{\gamma}$ and μ are hyper-parameters completely depending on the choice of a user. But, they can be intuitively estimated using some information. Let $r_{min} = \min_{l,l' \in N: l \neq l'} \|\mathbf{x}_l - \mathbf{x}_{l'}\|_p$ and $r_{max} = \max_{l,l' \in N: l \neq l'} \|\mathbf{x}_l - \mathbf{x}_{l'}\|_p$. It is also known that $0 < \underline{\gamma} \leq r \leq \bar{\gamma} \leq r_{max}$ from the way of generating radius candidates. r_{min} and r_{max} can give the rational instruction for setting of the hyper-parameters. For example, if the desirable volume of a norm ball should not be large, then one of the possible options is $\bar{\gamma} = 0.5 * r_{max}$. When norm balls covering only single point are needed, it is reasonable to select $\underline{\gamma} = 0.5 * r_{min}$. μ also can be

realized by setting $\mu = 0.5 * \bar{\gamma} = 0.5 * (0.5 * r_{max})$.

Eventually, the survivors out of $O(|N|^2)$ norm balls to satisfy the two conditions became the final candidates. The following pseudo-code shows norm ball generation algorithm where N_B is the number of norm balls and B_{cand} is the set of norm ball candidates.

Algorithm 1 Algorithm for final norm ball candidates

$N_B, r := 0$;
$B_{cand} := \varnothing$;
foreach $\mathbf{x}_l$ in $\{1, \ldots, |N|\}$ **do**
  foreach $\mathbf{x}_{l'}$ in $\{1, \ldots, |N|\}$ **do**
    $r := \|\mathbf{x}_l - \mathbf{x}_{ll}\|_p$;
    if $\underline{\gamma} \leq r \leq \bar{\gamma}$ **then**
      add $b_i = (\mathbf{c}_i, r_i) = (\mathbf{x}_l, r)$ to B_{cand}; and $N_B := N_B + 1$;
    end if
  end for
end for
foreach b_i in $\{1, \ldots, N_B\}$ **do**
  $X_p = \varnothing$;
  foreach $\mathbf{x}_l$ in $\{1, \ldots, |N|\}$ **do**
    if $\|\mathbf{c}_i - \mathbf{x}_l\|_p \leq r_i$ **then**
      add $\mathbf{x}_l$; to X_p;
    end if
  end for
  compute the center of mass of $X_p, \mathbf{c}_M$;
  if $\|\mathbf{c}_i - \mathbf{c}_M\|_p > \mu$ **then**
    eliminate b_i from B_{cand};
    $N_B := N_B - 1$;
  end if
end for

2.2.2 Selection of Norm Balls

The notations for norm balls and target objects are identical in the previous sections throughout this section. The final norm ball candidates are obtained from the **Algorithm 1**. The next step is to determine which norm balls are suitable for data description and build a classifier. In this section, a mathematical model for the selection of norm balls will be presented. Then, the meanings of the formulation are examined.

Using the final norm ball candidates in Section 2.2.1, the problem of building NBC, P_{NBC}, can be formulated as follows.

$$(P_{NBC}) \quad \min \quad \sum_{i \in B} y_i + C \sum_{l \in N} \xi_l \tag{2.1}$$

$$\text{s.t.} \quad \sum_{i \in B} w_l^i y_i \geq 1 - \xi_l, \quad \forall l \in N, \tag{2.2}$$

$$y_i \in \{0, 1\}, \quad \forall i \in B. \tag{2.3}$$

where $C > 0$ is a parameter. It is easily shown that P_{NBC} is associated to Set Covering Problem [5].

$y_i = 1$ indicates that norm ball candidate b_i is chosen, otherwise 0. In (2.2), w_l^i is computed before solving P_{NBC}: if $\mathbf{x}_l$ is in the final norm ball candidate b_i, $w_l^i = 1$, otherwise 0. ξ_l is introduced for an error corresponding to $\mathbf{x}_l$. Note that there is no need to add the condition for ξ_l like (2.3) because the left side of (2.2) forces ξ_ls to have 0 or 1.

The constraint (2.2) implies that $\mathbf{x}_l$ should be covered at least one of the final norm ball candidates, allowing ξ_l to violate the constraint. The objective

function (2.1) is to minimize the number of the selected norm balls which finally contributes to NBC and minimize the errors. The first term is related to the complexity of the model. The minimum norm balls guarantees high sparsity of the classifier.

Parameter C plays a role in controlling the trade-off between the complexity of the classifier and the training errors. C should be learned to produce the best classifier. It seems that there is a trouble in determining the value of C.

Also, C can be interpreted as the relative weight between the choice of a norm ball and error. Suppose that B includes $|N|$ norm balls covering each point $\mathbf{x}_l$ and a norm ball containing all of the points. Thus, C is meaningful when $\frac{1}{|N|} \leq C$.

When $C < \frac{1}{|N|}$, no norm balls are selected. If $y_i = 1$, then ξ_l such that $w_l^i = 1$ should be zero in order to minimize the objective function. It makes the objective value increase by 1. In contrast, the value does by $C < 1$ when $\xi_l = 1$. Since it is always expensive to choose a norm ball, the objective value equals $C \cdot |N| = \frac{1}{|N|} \cdot |N| < 1$.

In case of $C = \frac{1}{|N|}$, $\xi_l = 1$ is still preferred in the same context. Since $\sum_{i \in B} y_i + \frac{1}{|N|} \sum_{l \in N} \xi_l \geq 1$ from (2.2), the optimal objective value is 1, which has the two feasible solutions. One is to choose only one norm ball covering all the training data. The other is $y_i = 0 \ \forall i \in B$ and $\sum_{l \in N} \xi_l = |N|$.

As C grows, the benefit from violations decreases. When C becomes exactly 1, the difference between the profit of selecting a norm ball and that of making violation is zero. Then, we have the following propositions.

Proposition 1. *Assume that set the hyper-parameters* $(\underline{\gamma}, \bar{\gamma}, \mu)$ *such that each training point is covered by at least one norm ball. Let* $obj_{C=1}$ *and* $obj_{C>1}$ *be the optimal objective value of (2.1) when* $C = 1$ *and* $C > 1$ *respectively. Then, if the* P_{NBC} *is initialized by norm balls from* **Algorithm 1**, *then we have*

$$obj_{C=1} = obj_{C>1}$$

Proof. Let $(\mathbf{y}, \xi)$ be a feasible solution of P_{NBC}, and T be the set of these feasible solutions. It is clear that

$$\min_{(\mathbf{y},\xi)\in T}\{\sum_{i\in B} y_i + \sum_{l\in N} \xi_l\} \leq \min_{(\mathbf{y},\xi)\in T}\{\sum_{i\in B} y_i + C\sum_{l\in N} \xi_l\} \tag{2.4}$$

because C is a positive value such that $C > 1$ and all the variables are binary. It is sufficient to show that there exists a feasible solution which holds (2.4) with equality. We have two cases: Suppose that $(\mathbf{y}^*, \xi^*)$ is the optimal solution of the left-side of (2.4).

(a) When $\sum_{l\in N} \xi_l^* = 0$, the $(\mathbf{y}^*, \xi^*)$ is also optimal to the right-side of (2.4).

(b) When $\sum_{l\in N} \xi_l^* \neq 0$, let $N' \subseteq N$ is an index set such that $\xi_l^* = 1$ and B' be an index set of norm balls covering points $\mathbf{x}_{l'}$ $\forall l' \in N'$. Then, in $\mathbf{y}^*$, only change the component $y_{i'} = 0$ $\forall i' \in B'$ into $y_{i'} = 1$ $\forall i' \in B'$ and $\xi_{l'}^*$ $\forall l' \in N'$ becomes 0. The new solution makes (2.4) hold with equality.

□

If norm balls are initialized properly, then the case of $C > 1$ always has the similar solution to that of $C = 1$ except some violations. So, it is enough to consider the value of C from $1/|N|$ to number slightly bigger than 1.

In this formulation, the optimal data descriptors (norm balls) are obtained by solving an integer programming (IP) optimization problem similar to set covering problem. Similarly, [8] and [24] found hyper-rectangles (boxes) for data description from optimization problems such as P_{NBC} even in the context of binary classification. In both papers, the optimal solution is searched in the IP master problem by pricing the complex IP sub-problem whenever new box candidates are needed. Thus, there are some issues: initialization in the master problem, branching scheme and pricing the complicating sub-problem itself.

Compared to those formulations, P_{NBC} has more simple structure. It is possible to construct polynomial sized model which contains $|N|$ rows and $O(|N|^2)$ columns. Also, the solver just needs to pay attention to the master, and easily initialize columns representing norm ball candidates because a feasible solution *i.e.* a set of norm balls can be generated by controlling hyper-parameters related to the conditions in Section 2.2.1 if needed.

Figure 2.2 illustrates the overall procedure of building Norm Ball Classifier. The selection of p is similar to the choice of a kernel function like in SVDD and PDE. As seen in Figure 2.2, the four hyper-parameters are needed to be determined beforehand, but Section 2.2.1 and 2.2.2 shows that the users who builds NBC are able to deduce them.

Several characteristics are observed in NBC, compared to MoG and PDE. Suppose that the target class comes from multimodal Gaussian distribution and

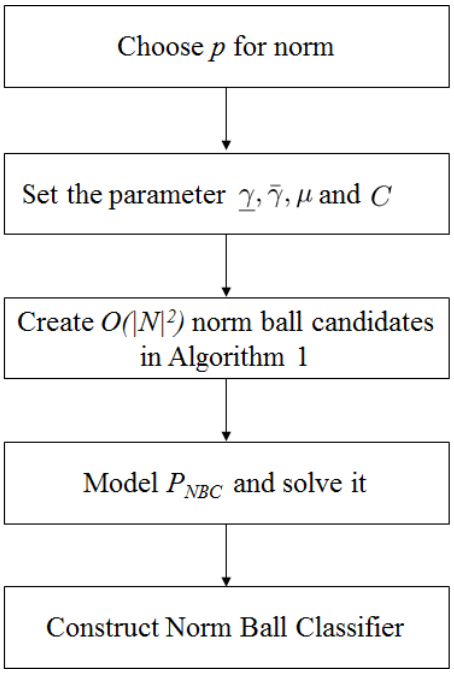


Figure 2.2: The overall procedure of the construction of NBC

2-norm ball classifier is constructed through the procedure. It seems that the norm balls in the classifier are the inverse image of the pdf *i.e.* regions with pdf value more than or equal to certain threshold. What our method does is that possible domains with desirable density are detected by **Algorithm 1** and the best ones which closely describe the target data are selected by solving the optimization problem. It is considered that if 1-norm is applied, the distribution of the target data is expected to approximately follow Laplacian distribution. In other words, our classifier is directly built in the original space where the training data exists instead of indirectly estimating pdf in density estimation methods.

Chapter 3

Experiments

In this chapter, we investigate the performance and behaviors of NBC through some experiments. Three experiments were conducted: preliminary test with artificial data, comparisons to other OCC classifiers using real data sets and noise test. The detailed results are reported in the following sections.

3.1 Experiment Design

3.1.1 Data

There are several artificial and real data sets from UCI Machine Learning Repository [16] in the experiments. The redundant rows were eliminated from the data sets before the experiments. Table 3.1 displays the characteristics of the data sets after pre-processing. In the fifth column of the table, the first value is the number of target data. Iris data has 3 classes: setosa, versicolor and virginica. Each class was used for training data and the rest of classes was regarded as non-target class data.

Bimodal, Hoof and Doughnut data were artificially generated. Bimodal data came from a distribution *i.e.* a mixture of two Gaussian distribution functions.

The non-target objects of Bimodal, Hoof and Doughnut data were randomly spreaded out in the real space. Especially, CSTH [31] is a simulation data set which represents a continuous stirred tank heat pilot plant. The practical generation of CSTH refers to [30].

Table 3.1: The characteristics of data sets in the experiments

type	data	Dimension	# class	# observations	source
Artificial	Bimodal	2	2	100/100	Generated
	Hoof	2	2	100/100	Generated
	Doughnut	2	2	100/100	Generated
Real	CSTH	3	2	100/100	Simulation
	Iris	4	3	50/50/49	UCI
	Liver	6	2	142/199	UCI
	WBC	9	2	213/237	UCI
	Ionosphere	34	2	225/125	UCI

3.1.2 NBCs and Comparison One-Class Classifier Group

In the experiment, we had NBC group and Comparison group. For NBC group, there were four norm ball classifiers using different types of norm: 1-norm, 2-norm, maximum norm and mixture of 1, 2 and maximum norm balls. From now, the NBCs using the 4 norm types will be called *one norm*, *two norm*, *max norm* and *mix norm* respectively in the results.

For comparison group, we selected some one-class classifiers mentioned in Section 1.2: MoG, PDE, SVDD, Nearest Neighbor Method (this method is briefly called 1-NN) and the extension of 1-NN using kth nearest neighbor (k-NN). These methods except MoG are non-parametric *i.e.* does not have the assumption of data distribution.

Also, they make use of the geometric information. PDE uses local density,

and 1-NN/k-NN classifies test objects by distance. SVDD finds a hypersphere which covers the target class using 2-norm distance. Although SVDD is not directly based on the concept of local density unlike PDE, it behaves similarly when Gaussian kernel $exp(-\frac{\|\mathbf{z}-\mathbf{x}\|_2^2}{2\sigma^2})$ is applied with small σ [28]. Since NBC is also a non-parametric and geometric way, it have in common with them. That is why these four methods were chosen for comparison group. Despite of parametric method, MoG was added because it has higher sparsity compared to the other techniques.

3.1.3 Experimental Settings

For NBC, there are four hyper-parameters: $\underline{\gamma}$, $\bar{\gamma}$, μ and C. $\underline{\gamma}$ was fixed in order to generate norm balls covering only single point by setting $\underline{\gamma} = 0.9 * r_{min}$. By the objective function (2.1) in Section 2.2.2, a norm ball with larger radius tends to be included more in the optimal solution to minimize the number of selected norm balls and errors. So, It was considered that $\bar{\gamma}$ would be more critical than $\underline{\gamma}$.

Let ρ_r, ρ_μ and ρ_C the control parameters of r_{max}, μ and C respectively. The following is the relation between a hyper-parameter and its control parameter.

$$\bar{\gamma} = r_{min} + \rho_r * (r_{max} - r_{min})$$

$$\mu = \rho_\mu * \bar{\gamma}$$

$$\rho_C = 1/|N| + \rho_C * (1 - 1/|N|)$$

ρ_r was varied such that $\rho_r = 0.01, 0.05, 0.1, 0.2, ..., 1$. ρ_μ was from $0.05, 0.1, 0.2 \sim$

1. Lastly, ρ_C ranged from $0.01, 0.1, 0.25, 0.5, 0.75, 1, 1.2$. With these parameters, each data has at least a norm ball candidate, **Proposition 1** can be applied. To see what happens when $C > 1$, $\rho_C = 1.2$ was added.

MoG has two hyper-parameters, the number of Gaussian functions and threshold ρ. The former ranged from 1 to 10 and the latter varied from e^{-25} to e^0. In case of PDE, Gaussian kernel was used where there were the two parameters: bandwidth $h = 2^{-6}, 2^{-5}, ..., 2^5, 2^6$ and threshold $\rho = e^{-25}, e^{-24}, ..., e^0$. SVDD utilized Radial Basis Function (RBF) kernel, $exp(-\gamma\|\mathbf{z} - \mathbf{x}\|_2^2)$. There are two parameters for $\gamma = 2^{-6}, 2^{-5}, ..., 2^5, 2^6$ and ν, the fraction of noises in the training data, which is related to regularization parameter such as C and equals $0.1, 0.2, ..., 1$. Lastly, in 1-NN/k -NN threshold ρ for d_1/d_2 and k should be decided. The quotient d_1/d_2 is very sensitive depending on the data, so a large value for the threshold should be considered. The threshold was set to increase by 0.1 from 0.1 to 4.5. k was from 2 to one-third of training points.

All the experiments followed the 10-fold cross validation rule except 5-fold cross validation rule for Iris data. The experiments was conducted on an Intel Core 3.10 GHz PC with 16GB RAM. P_{NBC} was solved by using Xpress solver [33]. The other classifiers in the comparison group was coded by PYTHON programming language employing the scikit-learn Library [20].

3.1.4 Performance Measure

In this experiment, F_1 score was used to evaluate a classifier. F_1 score is the harmonic mean of *recall* and *precision*:

$$F_1 = 2\frac{\text{recall} * \text{precision}}{\text{recall} + \text{precision}}$$

where recall is the fraction of the instances predicted as target by a classifier out of real target instances while precision is the fraction of the positive instances out of the instances predicted as positive. The better the classifier is, the more its score approximates 1.

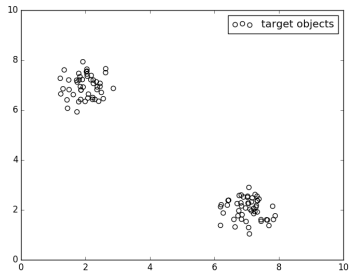
Even though there exist other performance measures, the reason why this performance was employed is that our classifier just determines whether an observation belongs to the target class or not. In other words, NBC does not have the concept of a threshold. Area Under Curve (AUC), which is frequently used as a performance measure, cannot be brought into our experiment. This is because it requires the classifier to score every test observation and the threshold should be inherent in the classifier itself. Furthermore, basically one-class classifier is constructed without knowing the exact distribution of non-target class. In such data imbalance situation, F_1 score can be a rational choice in [21].

3.2 Computational Results

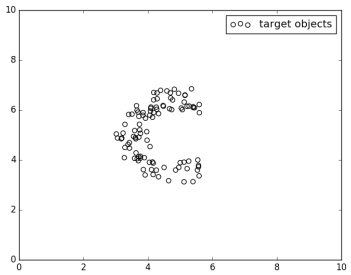
The classifiers are evaluated from the highest average F_1-score in the test corresponding to the selected classifiers which has the highest values of average F_1-score during the validation process.

3.2.1 Artificial Data

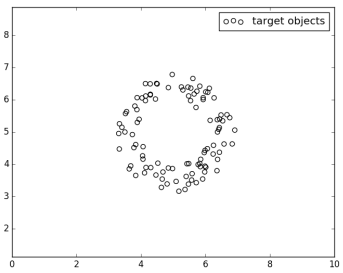
In this section, we deal with Bimodal, Hoof and Doughnut data. They were chosen to see how NBCs would work in different shaped data sets, especially non-convex ones. Figure 3.1 illustrates the three data distribution.



(a) Bimodal



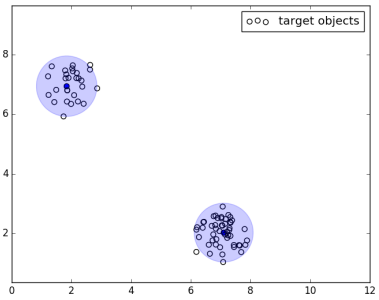
(b) Hoof



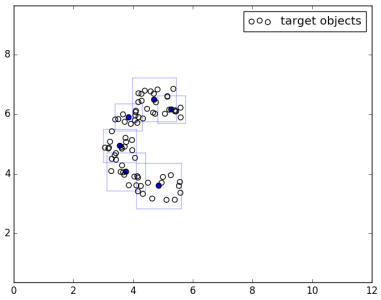
(c) Doughnut

Figure 3.1: Artificial data

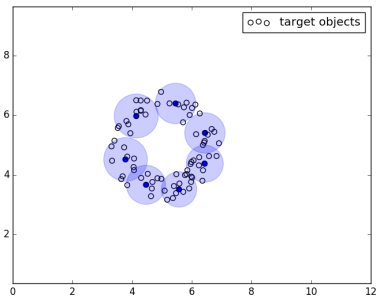
The results are summarized in Figure 3.2 and Table 3.2. Figure 3.2 displays how norm ball classifiers which have the best average F_1-score in the test describe the data. For Bimodal and Doughnut data, 2-norm ball classifier showed the best performance while for Hoof data, maximum norm ball classifier did. Within the comparison group, MoG or PDE was the best classifier whereas 1-NN was poorly performed because the reference sample is only one.



(a) Bimodal



(b) Hoof



(c) Doughnut

Figure 3.2: The best NBC for each data: the circles colored in dark blue are the center of each norm ball.

Table 3.2: The average recall, precision and F_1-score of artificial data sets (standard deviation in parentheses)

Bimodal									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.992(0.0)	1.0(0.0)	0.996(0.0)	0.91(0.16)	1.0(0.0)	0.944(0.1)	0.98(0.06)	1.0(0.0)	**0.989(0.03)**
max norm	0.999(0.0)	1.0(0.0)	0.999(0.0)	0.98(0.06)	0.974(0.05)	0.975(0.04)	0.96(0.07)	0.972(0.06)	0.965(0.06)
one norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.97(0.06)	0.963(0.06)	0.965(0.05)	0.96(0.07)	0.961(0.07)	0.96(0.06)
mix norm	0.998(0.0)	1.0(0.0)	0.999(0.0)	0.99(0.03)	0.974(0.05)	0.981(0.03)	1.0(0.0)	0.974(0.05)	0.986(0.03)
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.99(0.03)	0.974(0.05)	0.981(0.03)	0.96(0.09)	0.964(0.06)	0.959(0.06)
PDE	0.98(0.04)	1.0(0.0)	0.989(0.02)	0.98(0.04)	0.973(0.06)	0.976(0.04)	0.98(0.04)	0.973(0.06)	**0.976(0.04)**
1-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.91(0.09)	0.923(0.07)	0.913(0.07)	0.93(0.05)	0.924(0.07)	0.926(0.05)
k-NN	0.998(0.01)	1.0(0.0)	0.999(0.0)	0.98(0.06)	0.965(0.06)	0.97(0.04)	0.95(0.1)	0.911(0.07)	0.925(0.06)
SVDD	0.96(0.02)	1.0(0.0)	0.98(0.01)	0.89(0.14)	0.979(0.04)	0.926(0.09)	0.92(0.07)	0.978(0.04)	0.947(0.06)
Hoof									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.988(0.01)	1.0(0.0)	0.994(0.0)	0.97(0.05)	0.947(0.13)	0.954(0.09)	0.94(0.07)	0.875(0.16)	0.896(0.09)
max norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.95(0.07)	0.937(0.07)	0.941(0.04)	0.96(0.05)	0.922(0.08)	**0.938(0.04)**
one norm	0.992(0.0)	1.0(0.0)	0.996(0.0)	0.94(0.07)	0.942(0.12)	0.935(0.07)	0.93(0.08)	0.933(0.13)	0.922(0.07)
mix norm	0.994(0.01)	1.0(0.0)	0.997(0.0)	0.96(0.07)	0.928(0.08)	0.942(0.06)	0.94(0.08)	0.915(0.09)	0.923(0.06)
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.99(0.03)	0.93(0.06)	0.958(0.03)	0.96(0.05)	0.913(0.07)	0.933(0.04)
PDE	0.99(0.03)	1.0(0.0)	0.995(0.02)	0.99(0.03)	0.939(0.07)	0.962(0.03)	0.99(0.03)	0.947(0.06)	**0.967(0.03)**
1-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.85(0.14)	0.896(0.09)	0.864(0.08)	0.86(0.09)	0.906(0.07)	0.876(0.05)
k-NN	0.954(0.02)	1.0(0.0)	0.976(0.01)	0.96(0.05)	0.924(0.08)	0.938(0.04)	0.92(0.09)	0.938(0.1)	0.922(0.06)
SVDD	0.936(0.02)	1.0(0.0)	0.967(0.01)	0.9(0.08)	0.929(0.15)	0.907(0.09)	0.91(0.07)	0.952(0.1)	0.926(0.06)
Doughnut									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.973(0.01)	1.0(0.0)	0.986(0.0)	0.92(0.1)	0.921(0.09)	0.915(0.06)	0.9(0.09)	0.89(0.08)	**0.891(0.06)**
max norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.93(0.09)	0.897(0.1)	0.908(0.07)	0.89(0.11)	0.881(0.11)	0.877(0.07)
one norm	0.993(0.01)	1.0(0.0)	0.996(0.01)	0.89(0.11)	0.924(0.1)	0.898(0.08)	0.82(0.1)	0.915(0.08)	0.862(0.07)
mix norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.98(0.04)	0.805(0.13)	0.878(0.08)	0.92(0.1)	0.801(0.12)	0.847(0.07)
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.98(0.06)	0.937(0.07)	0.957(0.05)	0.98(0.04)	0.947(0.06)	**0.961(0.03)**
PDE	0.96(0.05)	1.0(0.0)	0.979(0.03)	0.96(0.05)	0.945(0.06)	0.951(0.04)	0.96(0.05)	0.946(0.06)	0.951(0.03)
1-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.88(0.06)	0.828(0.09)	0.849(0.06)	0.93(0.1)	0.809(0.08)	0.861(0.07)
k-NN	0.955(0.04)	1.0(0.0)	0.977(0.02)	0.98(0.04)	0.932(0.08)	0.953(0.05)	0.92(0.07)	0.928(0.07)	0.92(0.03)
SVDD	0.934(0.01)	1.0(0.0)	0.966(0.01)	0.88(0.12)	0.94(0.06)	0.904(0.07)	0.85(0.12)	0.946(0.07)	0.891(0.09)

The bolded value is the best within the group

In Bimodal data, 2-norm ball classifier covered the two separate clusters as expected. Overall, all the classifiers also had good performance. 1-NN and k-NN performed similarly because the nearest neighbor was close enough to each point, so 1-NN constructed the tight decision boundary and it was not necessary to refer more than 1 nearest neighbors. In fact, the best parameter k of k-NN was 3, which was a very small.

For Hoof data, maximum norm ball classifier was not as much as PDE, but NBCs was comparable to the comparison group. Compared to the others, Doughnut data was difficult to describe as all the classifiers have the lower score than one in the other two data sets. The best performance NBC, 2-norm ball classifier, which has lower F_1-score than that of MoG, PDE and k-NN, but its recall and precision are quite similar. Although MoG performed better than PDE, the difference of MoG between its recall and precision was bigger than that of PDE.

Table 3.3: Sparsity of classifiers: average number of non-zero terms corresponding to the best performance of parameter combination for artificial data sets (standard deviations in parentheses)

clasifiers	Bimodal	Hoof	Doughnut
two norm	2(0.0)	3.1(0.3)	7(0.0)
max norm	2(0.0)	5(0.0)	4(0.0)
one norm	2.2(0.4)	3(0.0)	6(0.0)
mix norm	2(0.0)	5.4(0.49)	6(0.0)
MoG*	4	9	5
PDE†	$\|N\|$	$\|N\|$	$\|N\|$
1-NN†	$\|N\|$	$\|N\|$	$\|N\|$
k-NN†	$\|N\|$	$\|N\|$	$\|N\|$
SVDD	6.2(0.6)	9.6(1.37)	10.1(0.7)

* The number of Gaussian functions is pre-specified beforehand by parameter.
† The model has the non-zero terms as many as all the training points.

Table 3.3, shows the number of non-zero terms related to the sparsity of the classifiers in the experiment. PDE, 1-NN and k-NN have all training points in their models, so $|N|$ represents the face in the Table 3.3. The number of norm balls and that of support vectors stand for the sparsity for NBC and SVDD respectively. The optimal parameter related to the number of Gaussian functions in MoG was four not two for Bimodal data unlike our expectations. This is because only with the given data the MoG classifier cannot exactly recognize that they were generated from two Gaussian functions. Also, it is observed that NBCs was completely better than the other classifiers in terms of sparsity. In Hoof and Doughnut data the NBCs required more norm balls to describe the non-convex sets.

3.2.2 Real Data

We conducted the experiment using real data including CSTH, UCI data. As seen in Figure 3.3, the two clusters are separately distributed in CSTH data. This data was indcluded in order to see whether NBC still performs well in the higher than two dimensional space.

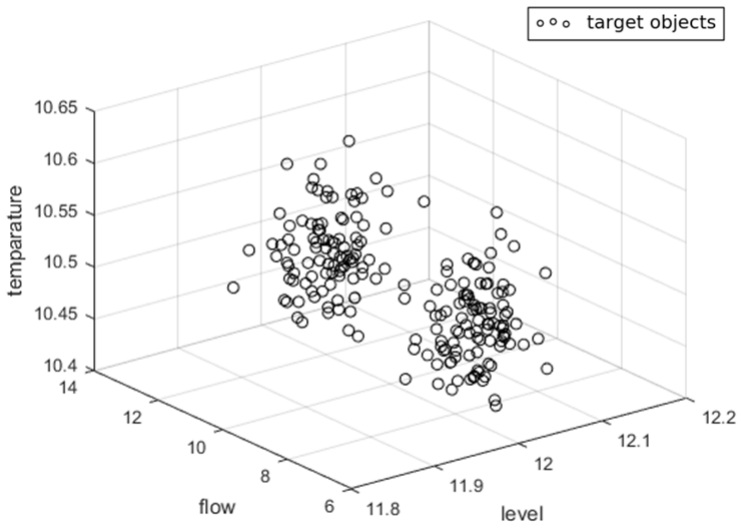


Figure 3.3: CSTH data

Table 3.4, 3.5 and 3.6 displays the average recall, precision and F_1-score of real data sets. Thoughout training process, all the classifiers handled target class objects, so the precisions were 1. In the comparison group, MoG and PDE generally performed better than the other classifiers and 1-NN did poorly except Liver data. The four NBCs was comparable to the other classifiers in the comparison group. In CSTH and Iris-setosa data, the performance of SVDD was relatively lower showing the big difference between training and test in terms

of recall. SVDD behaved badly rather in the well-distributed data sets.

The first distinctive point is that F_1-score of all the classifiers in Liver data was much lower than in other data sets: the recall values were close to 1 while precision ones was meaningless, which is lower than 0.5. This gap implies that the classifiers accepted many of non-target objects. Especially, SVDD poorly performed in all the processes, which leads to underfitting. It seems that pre-processing such as scaling is needed before applying SVDD technique. PDE had the lower F_1-score, which means that it is very difficult to estimate the overall pdf by considering all the pdfs of each point and there are no patterns in the distribution. Rather, MoG technique using small number Gaussian functions showed better performance. However, the trend of the recall in PDE fluctuated less compared to MoG. Instead, referring to very local density information was better in case of k-NN. In fact, Liver data is known to belong to hard-class data for classification even in the context of multi-class classification area. Despite this data characteristics, all the four NBCs showed better performances.

Secondly, maximum norm ball classifier had higher precision than recall rate except Liver data. It means that using the maximum norm, NBC tends to have more strict the acceptance decision level regardless of data. In contrast, it is observed that mix norm ball classifier had higher recall rate in Iris-versicolor, Iris-virginica and Ionosphere data. In these 3 data, average test F_1 score was the lowest and its standard deviation is highest among NBCs and their F_1 scores in the training approximated 1. This suggests that there is a chance of overfitting when using mixture of norm balls due to many choices of norm balls to fit the data. In addition, with high C mix norm ball classifier was forced to select norm

balls with bigger radius to minimize the objective function in P_{NBC}.

Lastly, all the classifiers except mix norm classifier tends to accept objects rather than reject them in Ionoshpere data with high dimensional space. As the dimension becomes higher, the number of space where an objects can be place exponentially increases. It is likely that even though a classifier tightly describes the data, there is a limit to detect non-target class objects. It seems that NBCs still showed good performance even in the high dimensional space.

From Table 3.7, it can be easily found that NBCs typically had higher sparsity than the classifiers in comparison group. Compared to MoG, NBCs performed comparably with high performance and sparsity. In case of Iris-versicolor, the number of the selected mix norm balls was more than 10. This is because the classifier had the high penalty of violations with $C = 1$ but the with small maximum radius parameter $\rho_r = 0.4$. In result, it is figured out that NBCs is quite comparable to the other classifiers in comparison group with relatively higher sparsity.

Table 3.4: The average recall, precision and F_1-score of CSTH, Iris-setosa and Iris-versicolor data (standard deviation in parentheses)

CSTH									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.984(0.01)	1.0(0.0)	0.992(0.0)	0.91(0.16)	1.0(0.0)	0.944(0.1)	0.98(0.06)	1.0(0.0)	0.989(0.03)
max norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.88(0.2)	1.0(0.0)	0.922(0.14)	0.91(0.12)	1.0(0.0)	0.948(0.07)
one norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.88(0.2)	1.0(0.0)	0.922(0.13)	0.99(0.03)	1.0(0.0)	**0.995(0.02)**
mix norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.88(0.2)	1.0(0.0)	0.922(0.14)	0.91(0.12)	1.0(0.0)	0.948(0.07)
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
PDE	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
1-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.92(0.16)	1.0(0.0)	0.95(0.1)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
k-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
SVDD	0.992(0.02)	1.0(0.0)	0.996(0.01)	0.87(0.3)	0.9(0.3)	0.884(0.3)	0.88(0.21)	0.938(0.19)	0.906(0.2)
Iris-setosa									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.987(0.02)	1.0(0.0)	0.993(0.01)	0.98(0.04)	1.0(0.0)	0.989(0.02)	0.98(0.04)	1.0(0.0)	0.989(0.02)
max norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.98(0.04)	1.0(0.0)	0.989(0.02)	0.98(0.04)	1.0(0.0)	0.989(0.02)
one norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.92(0.12)	1.0(0.0)	0.954(0.07)
mix norm	0.993(0.01)	1.0(0.0)	0.997(0.01)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
PDE	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
1-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
k-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
SVDD	0.987(0.03)	1.0(0.0)	0.993(0.01)	0.94(0.08)	1.0(0.0)	0.967(0.04)	0.94(0.08)	1.0(0.0)	0.967(0.04)
Iris-versicolor									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.987(0.02)	1.0(0.0)	0.993(0.01)	0.9(0.09)	0.934(0.09)	0.909(0.05)	0.88(0.12)	0.828(0.17)	0.837(0.1)
max norm	0.993(0.01)	1.0(0.0)	0.997(0.01)	0.88(0.08)	0.905(0.12)	0.883(0.05)	0.84(0.08)	0.859(0.15)	0.845(0.11)
one norm	0.973(0.02)	1.0(0.0)	0.986(0.01)	0.98(0.04)	0.917(0.08)	0.944(0.04)	0.9(0.11)	0.851(0.12)	**0.867(0.09)**
mix norm	0.993(0.01)	1.0(0.0)	0.997(0.01)	1.0(0.0)	0.781(0.13)	0.871(0.09)	0.96(0.05)	0.752(0.21)	0.83(0.16)
MoG	0.993(0.01)	1.0(0.0)	0.997(0.01)	0.98(0.04)	0.925(0.11)	0.947(0.06)	0.96(0.05)	0.863(0.13)	0.903(0.08)
PDE	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.902(0.09)	0.946(0.05)	0.98(0.04)	0.854(0.15)	**0.907(0.1)**
1-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.86(0.1)	0.83(0.13)	0.83(0.04)	0.9(0.15)	0.772(0.2)	0.796(0.11)
k-NN	0.973(0.02)	1.0(0.0)	0.986(0.01)	0.96(0.05)	0.838(0.13)	0.887(0.06)	0.96(0.05)	0.799(0.17)	0.863(0.11)
SVDD	0.94(0.04)	1.0(0.0)	0.969(0.02)	0.9(0.11)	0.892(0.1)	0.891(0.08)	0.86(0.1)	0.9(0.13)	0.871(0.08)

The bolded value is the best within the group

Table 3.5: The average recall, precision and F_1-score of Iris-virginica, Liver, WBC data (standard deviation in parentheses)

Iris-virginica									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.918(0.03)	1.0(0.0)	0.957(0.02)	0.9(0.09)	0.884(0.11)	0.882(0.04)	0.84(0.15)	0.808(0.16)	0.799(0.06)
max norm	0.953(0.02)	1.0(0.0)	0.976(0.01)	0.9(0.09)	0.922(0.07)	0.907(0.06)	0.86(0.15)	0.871(0.12)	**0.846(0.06)**
one norm	0.973(0.03)	1.0(0.0)	0.986(0.01)	0.9(0.13)	0.875(0.2)	0.861(0.12)	0.838(0.08)	0.866(0.13)	0.84(0.04)
mix norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.738(0.26)	0.822(0.19)	0.98(0.04)	0.739(0.32)	0.795(0.23)
MoG	0.973(0.03)	1.0(0.0)	0.986(0.01)	0.938(0.05)	0.874(0.16)	0.899(0.1)	0.94(0.05)	0.849(0.19)	0.879(0.11)
PDE	0.96(0.08)	1.0(0.0)	0.978(0.04)	0.96(0.08)	0.864(0.13)	0.9(0.06)	0.94(0.08)	0.869(0.14)	**0.892(0.07)**
1-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.9(0.15)	0.706(0.24)	0.75(0.13)	0.8(0.18)	0.777(0.2)	0.76(0.13)
k-NN	0.871(0.04)	1.0(0.0)	0.931(0.02)	0.86(0.08)	0.812(0.16)	0.822(0.07)	0.82(0.1)	0.752(0.21)	0.761(0.1)
SVDD	0.912(0.02)	1.0(0.0)	0.954(0.01)	0.86(0.1)	0.96(0.05)	0.901(0.05)	0.84(0.14)	0.927(0.09)	0.869(0.07)
Liver									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.908(0.09)	1.0(0.0)	0.95(0.05)	0.929(0.08)	0.445(0.05)	0.601(0.05)	0.937(0.06)	0.439(0.04)	**0.596(0.04)**
max norm	0.984(0.01)	1.0(0.0)	0.992(0.0)	0.979(0.03)	0.425(0.01)	0.593(0.02)	0.972(0.05)	0.425(0.02)	0.591(0.02)
one norm	0.877(0.02)	1.0(0.0)	0.934(0.01)	0.837(0.16)	0.47(0.05)	0.597(0.07)	0.823(0.18)	0.464(0.05)	0.587(0.08)
mix norm	0.984(0.01)	1.0(0.0)	0.992(0.0)	0.986(0.03)	0.427(0.01)	0.596(0.01)	0.986(0.03)	0.427(0.01)	0.596(0.01)
MoG	0.938(0.01)	1.0(0.0)	0.968(0.01)	0.867(0.12)	0.448(0.04)	0.589(0.05)	0.86(0.12)	0.448(0.04)	**0.588(0.06)**
PDE	0.81(0.16)	1.0(0.0)	0.885(0.11)	0.81(0.16)	0.435(0.04)	0.563(0.07)	0.802(0.16)	0.432(0.04)	0.559(0.07)
1-NN	0.998(0.0)	1.0(0.0)	0.999(0.0)	0.958(0.05)	0.428(0.03)	0.591(0.03)	0.951(0.04)	0.428(0.02)	0.59(0.02)
k-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.993(0.02)	0.429(0.02)	0.599(0.02)	0.993(0.02)	0.428(0.02)	**0.598(0.02)**
SVDD	0.427(0.01)	1.0(0.0)	0.598(0.01)	0.274(0.16)	0.534(0.14)	0.346(0.17)	0.289(0.17)	0.557(0.19)	0.364(0.19)
WBC									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.947(0.01)	1.0(0.0)	0.973(0.0)	0.953(0.05)	0.948(0.06)	0.95(0.05)	0.93(0.07)	0.943(0.06)	0.934(0.05)
max norm	0.939(0.01)	1.0(0.0)	0.969(0.0)	0.93(0.05)	0.942(0.05)	0.935(0.04)	0.93(0.05)	0.942(0.05)	0.935(0.04)
one norm	0.948(0.02)	1.0(0.0)	0.973(0.01)	0.958(0.06)	0.956(0.04)	0.956(0.04)	0.944(0.07)	0.945(0.04)	**0.943(0.05)**
mix norm	0.921(0.02)	1.0(0.0)	0.959(0.01)	0.92(0.06)	0.942(0.05)	0.929(0.04)	0.93(0.05)	0.946(0.05)	0.937(0.04)
MoG	0.999(0.0)	1.0(0.0)	1.0(0.0)	0.902(0.07)	0.962(0.05)	0.929(0.05)	0.864(0.08)	0.97(0.04)	0.912(0.06)
PDE	0.92(0.07)	1.0(0.0)	0.957(0.04)	0.92(0.07)	0.962(0.04)	0.939(0.05)	0.92(0.07)	0.962(0.04)	**0.939(0.05)**
1-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.859(0.07)	0.567(0.07)	0.682(0.07)	0.854(0.07)	0.552(0.05)	0.67(0.05)
k-NN	0.941(0.01)	1.0(0.0)	0.97(0.01)	0.944(0.04)	0.904(0.05)	0.923(0.03)	0.939(0.06)	0.877(0.08)	0.905(0.05)
SVDD	0.959(0.01)	1.0(0.0)	0.979(0.0)	0.912(0.06)	0.922(0.06)	0.916(0.05)	0.921(0.05)	0.907(0.07)	0.913(0.06)

The bolded value is the best within the group

Table 3.6: The average recall, precision and F_1-score of Ionosphere data (standard deviation in parentheses)

Ionosphere									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.991(0.0)	1.0(0.0)	0.996(0.0)	0.982(0.03)	0.908(0.07)	0.942(0.04)	0.973(0.03)	0.901(0.06)	0.935(0.04)
max norm	0.956(0.01)	1.0(0.0)	0.977(0.01)	0.934(0.04)	0.961(0.04)	0.947(0.03)	0.925(0.04)	0.969(0.03)	**0.946(0.03)**
one norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.978(0.02)	0.899(0.06)	0.935(0.03)	0.969(0.03)	0.869(0.06)	0.915(0.05)
mix norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.991(0.03)	0.77(0.09)	0.863(0.05)	0.996(0.01)	0.763(0.09)	0.861(0.05)
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.99(0.03)	0.93(0.06)	0.958(0.03)	0.96(0.05)	0.913(0.07)	0.933(0.04)
PDE	0.961(0.05)	1.0(0.0)	0.979(0.02)	0.961(0.05)	0.932(0.06)	0.944(0.03)	0.96(0.04)	0.932(0.06)	**0.944(0.03)**
1-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.938(0.04)	0.785(0.08)	0.853(0.06)	0.92(0.06)	0.764(0.06)	0.832(0.04)
k-NN	0.928(0.02)	1.0(0.0)	0.963(0.01)	0.9(0.13)	0.882(0.07)	0.883(0.08)	0.878(0.16)	0.843(0.07)	0.854(0.1)
SVDD	0.97(0.01)	1.0(0.0)	0.984(0.0)	0.921(0.08)	0.902(0.07)	0.908(0.05)	0.926(0.07)	0.899(0.07)	0.909(0.05)

The bolded value is the best within the group

Table 3.7: Sparsity of classifiers: average number of non-zero terms corresponding to the best performance of parameter combination for real data sets (standard deviations in parentheses)

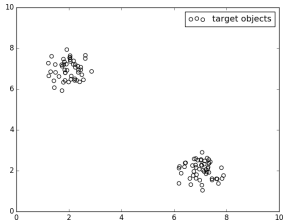
clasifiers	CSTH	Iris-setosa	Iris-versicolor	Iris-virginica	Liver	WBC	Ionosphere
two norm	2(0.0)	1(0.0)	2(0.0)	1.2(0.4)	1.2(0.4)	1(0.0)	3(0.0)
max norm	2(0.0)	1(0.0)	3.83(0.8)	2.2(0.4)	1(0.0)	1(0.0)	2.4(0.66)
one norm	2(0.0)	1(0.0)	2.2(0.4)	2(0.63)	1.9(0.3)	1.11(0.31)	2.2(0.4)
mix norm	2(0.0)	1(0.0)	14.6(3.38)	4.6(0.5)	1(0.0)	1.9(0.3)	1(0.0)
MoG*	2	1	1	1	2	5	9
PDE†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
1-NN†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
k-NN†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
SVDD†	2.1(0.3)	2(0.0)	3.4(1.2)	4.8(0.75)	83(2.68)	13.5(1.02)	11.7(1.1)

* The number of Gaussian functions is pre-specified beforehand by parameter.
† The model has the non-zero terms as many as all the training points.

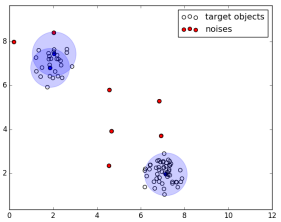
3.2.3 Noise Experiment

In this section, we investigate the NBCs behavior with noises. The data used in this experiment is Bimodal, CSTH and UCI data sets. While training, some non-target objects were included with target class label. In the process of validation and test, they recovered their original labels. The noise level increased from 5 to 20% in the training set. For each data, the performance of the classifier was computed depending on the noise level.

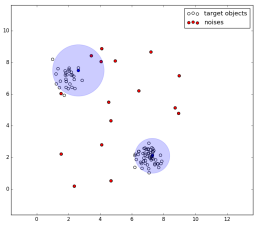
First, take a look at Bimodal data which can be examined in the low dimensional space. 2-norm ball classifiers generally outperformed the other NBCs as expected. Figure 3.4 illustrates that as the noise level increases, the region which accepts objects becomes larger.



(a) 0%



(b) 10%



(c) 20%

Figure 3.4: 2-norm ball classifier when noise 0, 10 and 20% for Bimodal data: the circles colored in dark blue are the center of each norm ball.

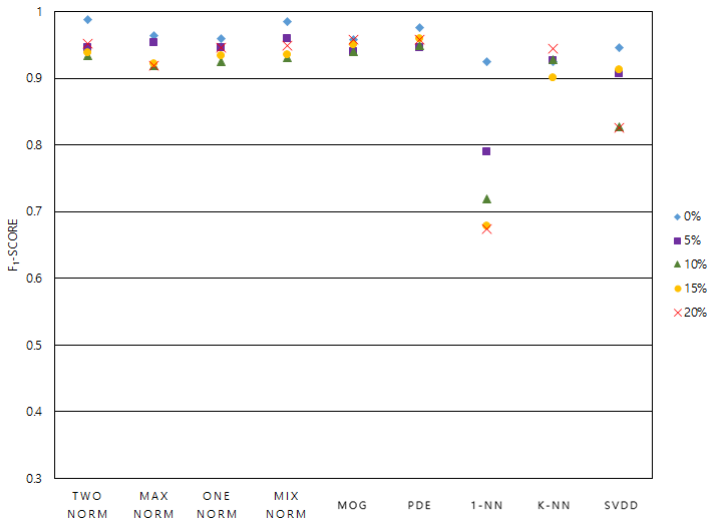
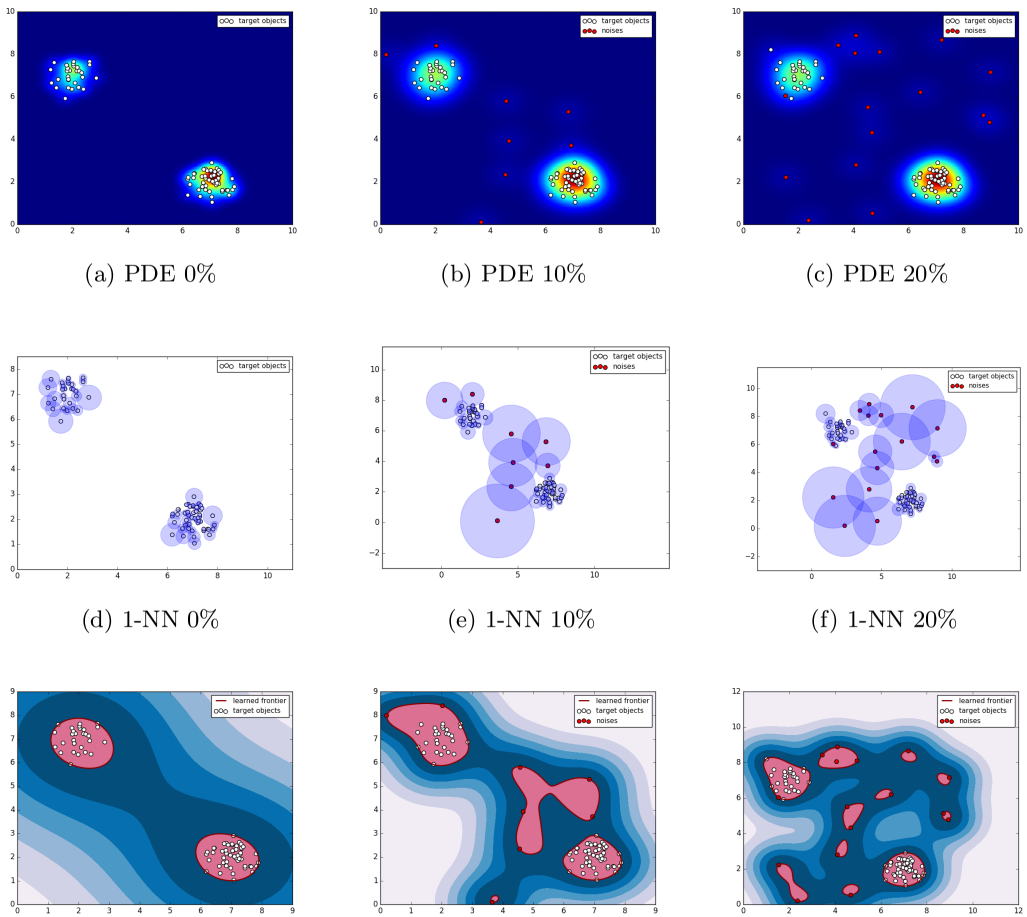


Figure 3.5: The average test F_1 score of the best parameter combination in the validation depending on noise level for Bimodal data

The graph in Figure 3.5 explains the trend of the highest average test F_1 score on each noise level. Obviously, the score of 1-NN asymptomatically decreases with the largest spread from 0% to 20%. With some fluctuations, SVDD also has lower F_1 scores. However, NBCs including MoG, PDE and k-NN shows the stable scores regardless of noises level. The result follows our expectation that two norm ball classifiers would perform better out of NBCs. On noise level 20%, maximum norm ball classifier is below the other NBCs. The other hyperparameters were the same but the μ of maximum norm ball classifier was not as tight as that of other NBCs. In result, it constructed more unbalanced a norm ball covering more noises.

In Figure 3.6, it is shown how in the comparison group, PDE, 1-NN and SVDD behave graphically. The PDE results implies that the acceptance region



(a) PDE 0% (b) PDE 10% (c) PDE 20%

(d) 1-NN 0% (e) 1-NN 10% (f) 1-NN 20%

(g) SVDD 0% (h) SVDD 10% (i) SVDD 20%

Figure 3.6: PDE, 1-NN and SVDD classifier with noises

in two dimensional space is colored in light green and red the zone in dark blue has lower local density. PDE remains stable regardless of noise level while for 1-NN and SVDD, the acceptance region of the target class objects becomes larger. SVDD tends to fit the noises, but at least attempts to tighten the acceptance region. It is figured out that 1-NN contains the useless space because of noises by allowing non-target class objects to be accepted. Of course, we can directly recognize that 1-NN is highly affected by noises from these pictures.

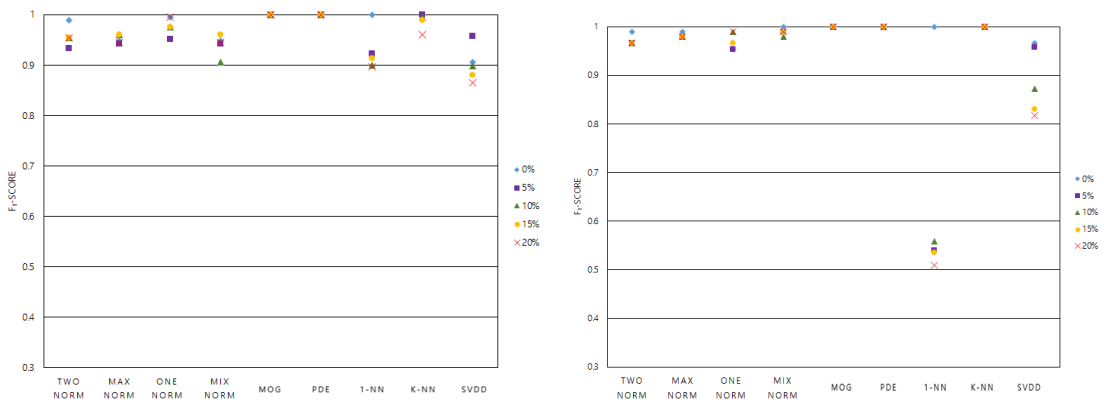
The other results for each data are summarized in Figure 3.7, 3.8 and Table 3.8, 3.9. Generally, MoG and PDE was robust to noises. 1-NN was very sensitive to them except liver data where 1-NN and k-NN had higher score because of the same reason in the Section 3.2.3. It is observed that SVDD performs poorly compared to the other 7 classifiers. This is because the experiment was conducted without scaling.

In the high dimensional data such as WBC and Ionosphere data sets, as the noise level grows, the MoG became conservative to acceptance of target objects with higher precision than recall value and the gap between these values was bigger. On the other hands, PDE behaved in an opposite way with higher recall than precision value. It is demonstrated from the fact that the MoG had F_1 score of training data higher than that of test, which leads to the overfitting to training data with less F_1 score compared to PDE. This is because PDE estimates more density information from all the training points whereas there is a limitation of the number of Gaussians functions and MoG should use the information even from noises.

NBCs show the comparable results to MoG and PDE with some fluctuations. The most interesting point was that among NBCs, maximum norm ball classifier was the least sensitive to noises with a low deviation of the best average F_1-scores in the test. On the contrary, mix norm ball classifier in high dimensional space fluctuated most with biggest F_1-score gap between training and test. It seems that mix norm balls have a lot of norm ball combinations for covering the data rather resulting in overfitting the training data. Also, in the high dimensional space, each norm ball has the larger number of space combination

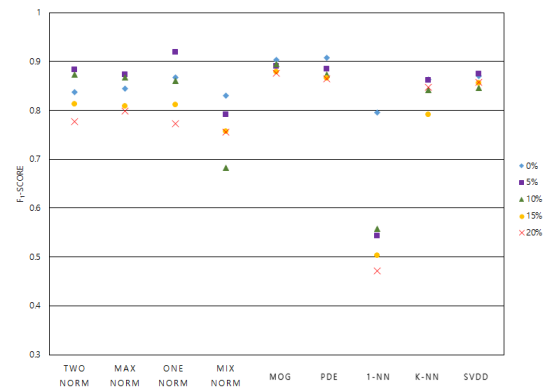
and the sparse space. In Ionosphere data, mix norm ball classifier performed poorly compared to the other NBCs.

From Table 3.10 and 3.11 apparently show the NBCs advantages, high sparsity of the model even if noises disturbs the perform of them. Relatively, MoG showed higher sparsity than NBCs. Some results with higher many norm balls in NBCs means that the classifier consisted of many norm balls with higher best C value, which requires more coverage of training data and leads to overfitting.

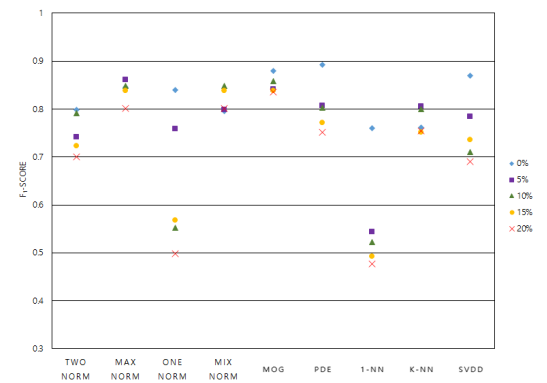


(a) CSTH

(b) Iris-setosa

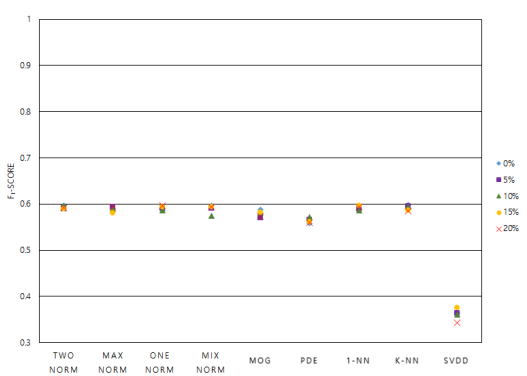


(c) Iris-versicolor

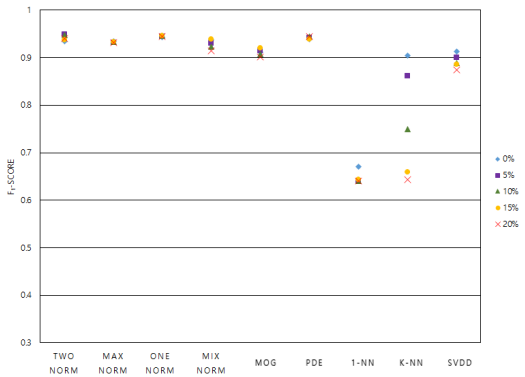


(d) Iris-virginica

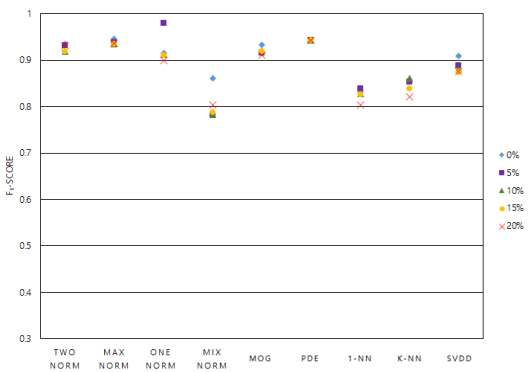
Figure 3.7: The average F_1 score in the test depending on noise level for CSTH, Iris-setosa, Iris-versicolor and Iris-virginica data



(a) Liver



(b) WBC



(c) Ionoshpere

Figure 3.8: The average F_1 score in the test depending on noise level for Liver, WBC and Ionosphere data

Table 3.8: The average F_1-score in the test on noise level of Bimodal, CSTH, Iris-setosa and versicolor data sets (standard deviation in parentheses)

Bimodal					
classifier	0%	5%	10%	15%	20%
two norm	**0.989(0.03)**	0.946(0.06)	**0.934(0.08)**	**0.939(0.06)**	**0.952(0.04)**
max norm	0.965(0.06)	0.954(0.05)	0.919(0.05)	0.923(0.07)	0.919(0.06)
one norm	0.96(0.06)	0.946(0.06)	0.925(0.07)	0.934(0.07)	0.947(0.05)
mix norm	0.986(0.03)	**0.96(0.05)**	0.932(0.06)	0.936(0.06)	0.95(0.05)
MoG	0.959(0.06)	0.94(0.07)	0.94(0.05)	0.951(0.05)	**0.958(0.04)**
PDE	**0.976(0.04)**	**0.946(0.05)**	**0.95(0.04)**	**0.96(0.05)**	**0.958(0.04)**
1-NN	0.926(0.05)	0.79(0.08)	0.719(0.07)	0.679(0.06)	0.675(0.05)
k-NN	0.925(0.06)	0.927(0.07)	0.929(0.04)	0.901(0.06)	0.945(0.04)
SVDD	0.947(0.06)	0.908(0.07)	0.828(0.1)	0.914(0.07)	0.826(0.07)
CSTH					
classifier	0%	5%	10%	15%	20%
two norm	0.989(0.03)	0.934(0.13)	0.954(0.07)	0.954(0.07)	0.954(0.07)
max norm	0.948(0.07)	0.942(0.07)	0.96(0.06)	**0.96(0.06)**	0.946(0.07)
one norm	**0.995(0.02)**	**0.952(0.13)**	**0.975(0.06)**	0.975(0.06)	**0.995(0.02)**
mix norm	0.948(0.07)	0.942(0.07)	0.906(0.15)	0.96(0.06)	0.946(0.07)
MoG	**1.0(0.0)**	**1.0(0.0)**	**1.0(0.0)**	**1.0(0.0)**	**1.0(0.0)**
PDE	**1.0(0.0)**	**1.0(0.0)**	**1.0(0.0)**	**1.0(0.0)**	**1.0(0.0)**
1-NN	**1.0(0.0)**	0.922(0.11)	0.9(0.11)	0.914(0.02)	0.897(0.02)
k-NN	**1.0(0.0)**	**1.0(0.0)**	0.995(0.01)	0.989(0.02)	0.961(0.06)
SVDD	0.906(0.2)	0.946(0.06)	0.899(0.09)	0.88(0.08)	0.865(0.09)
Iris-setosa					
classifier	0%	5%	10%	15%	20%
two norm	0.989(0.02)	0.967(0.04)	0.967(0.04)	0.967(0.04)	0.967(0.04)
max norm	0.989(0.02)	0.979(0.03)	0.979(0.03)	0.979(0.03)	0.979(0.03)
one norm	0.954(0.07)	0.954(0.07)	**0.989(0.02)**	0.967(0.04)	**0.989(0.02)**
mix norm	**1.0(0.0)**	**0.989(0.02)**	0.979(0.03)	**0.989(0.02)**	**0.989(0.02)**
MoG	1.0(0.0)	**1.0(0.0)**	**1.0(0.0)**	**1.0(0.0)**	**1.0(0.0)**
PDE	**1.0(0.0)**	**1.0(0.0)**	**1.0(0.0)**	**1.0(0.0)**	**1.0(0.0)**
1-NN	**1.0(0.0)**	0.54(0.06)	0.558(0.08)	0.536(0.07)	0.51(0.08)
k-NN	**1.0(0.0)**	**1.0(0.0)**	**1.0(0.0)**	**1.0(0.0)**	**1.0(0.0)**
SVDD	0.967(0.04)	0.958(0.04)	0.873(0.07)	0.83(0.08)	0.818(0.11)
Iris-versicolor					
classifier	0%	5%	10%	15%	20%
two norm	0.837(0.1)	0.883(0.1)	**0.873(0.09)**	**0.813(0.12)**	0.777(0.15)
max norm	0.845(0.11)	0.873(0.14)	0.868(0.11)	0.809(0.12)	**0.799(0.12)**
one norm	**0.867(0.09)**	**0.919(0.1)**	0.86(0.09)	0.812(0.1)	0.773(0.13)
mix norm	0.83(0.16)	0.791(0.16)	0.683(0.22)	0.757(0.17)	0.756(0.2)
MoG	0.903(0.08)	**0.89(0.1)**	**0.895(0.08)**	**0.879(0.07)**	**0.876(0.05)**
PDE	**0.907(0.1)**	0.885(0.11)	0.873(0.12)	0.866(0.11)	0.864(0.11)
1-NN	0.796(0.11)	0.544(0.12)	0.558(0.1)	0.503(0.01)	0.472(0.13)
k-NN	0.863(0.11)	0.862(0.14)	0.841(0.11)	0.791(0.07)	0.847(0.11)
SVDD	0.871(0.08)	0.875(0.08)	0.846(0.08)	0.856(0.07)	0.858(0.07)

The bolded value is the best within the group.

Table 3.9: The average test F_1-score on noise level of Iris-virginica, Liver, WBC and Ionosphere data sets (standard deviation in parentheses)

Iris-virginica					
classifier	0%	5%	10%	15%	20%
two norm	0.799(0.06)	0.741(0.13)	0.791(0.07)	0.723(0.14)	0.701(0.12)
max norm	**0.846(0.06)**	**0.861(0.1)**	**0.849(0.11)**	**0.838(0.1)**	**0.801(0.09)**
one norm	0.84(0.04)	0.758(0.08)	0.553(0.17)	0.568(0.11)	0.499(0.06)
mix norm	0.795(0.23)	0.798(0.11)	**0.849(0.11)**	**0.838(0.1)**	**0.801(0.09)**
MoG	0.879(0.11)	**0.841(0.12)**	**0.859(0.08)**	**0.839(0.1)**	**0.835(0.14)**
PDE	**0.892(0.07)**	0.807(0.06)	0.803(0.08)	0.771(0.05)	0.752(0.1)
1-NN	0.76(0.13)	0.544(0.17)	0.522(0.14)	0.493(0.04)	0.477(0.11)
k-NN	0.761(0.1)	0.806(0.08)	0.8(0.06)	0.751(0.07)	0.754(0.11)
SVDD	0.869(0.07)	0.785(0.08)	0.711(0.03)	0.736(0.09)	0.691(0.1)
Liver					
classifier	0%	5%	10%	15%	20%
two norm	**0.596(0.04)**	0.59(0.01)	**0.597(0.01)**	0.589(0.02)	0.592(0.02)
max norm	0.591(0.02)	**0.593(0.02)**	0.589(0.02)	0.58(0.03)	0.593(0.02)
one norm	0.587(0.08)	0.591(0.02)	0.587(0.01)	**0.594(0.03)**	**0.597(0.04)**
mix norm	**0.596(0.01)**	0.591(0.02)	0.575(0.03)	**0.594(0.01)**	0.593(0.02)
MoG	**0.588(0.06)**	0.57(0.06)	0.582(0.05)	0.582(0.06)	0.573(0.06)
PDE	0.559(0.07)	0.566(0.07)	0.572(0.06)	0.562(0.09)	0.56(0.09)
1-NN	0.59(0.02)	0.589(0.02)	0.587(0.03)	**0.596(0.02)**	**0.594(0.01)**
k-NN	**0.598(0.02)**	**0.595(0.02)**	**0.594(0.02)**	0.587(0.01)	0.585(0.04)
SVDD	0.364(0.19)	0.364(0.15)	0.363(0.15)	0.375(0.18)	0.343(0.15)
WBC					
classifier	0%	5%	10%	15%	20%
two norm	0.934(0.05)	**0.949(0.04)**	**0.947(0.04)**	0.937(0.05)	0.94(0.05)
max norm	0.935(0.04)	0.932(0.05)	0.933(0.04)	0.933(0.04)	0.932(0.04)
one norm	**0.943(0.05)**	0.944(0.04)	0.946(0.04)	**0.946(0.05)**	**0.945(0.04)**
mix norm	0.937(0.04)	0.931(0.04)	0.923(0.05)	0.939(0.04)	0.914(0.03)
MoG	0.912(0.06)	0.914(0.07)	0.907(0.06)	0.92(0.04)	0.902(0.05)
PDE	**0.939(0.05)**	**0.942(0.05)**	**0.944(0.04)**	**0.939(0.05)**	**0.944(0.05)**
1-NN	0.67(0.05)	0.641(0.01)	0.641(0.01)	0.641(0.01)	0.641(0.01)
k-NN	0.905(0.05)	0.862(0.06)	0.749(0.05)	0.659(0.1)	0.643(0.02)
SVDD	0.913(0.06)	0.9(0.04)	0.889(0.04)	0.886(0.04)	0.875(0.04)
Ionoshpere					
classifier	0%	5%	10%	15%	20%
two norm	0.935(0.04)	0.931(0.03)	0.918(0.06)	0.919(0.05)	0.932(0.04)
max norm	**0.946(0.03)**	0.938(0.04)	**0.935(0.03)**	**0.934(0.05)**	**0.936(0.04)**
one norm	0.915(0.05)	0.98(0.01)	0.912(0.05)	0.91(0.05)	0.899(0.04)
mix norm	0.861(0.05)	**0.783(0.01)**	0.783(0.0)	0.788(0.02)	0.804(0.04)
MoG	0.933(0.04)	0.915(0.05)	0.92(0.06)	0.919(0.06)	0.911(0.06)
PDE	**0.944(0.03)**	**0.943(0.04)**	**0.942(0.04)**	**0.943(0.05)**	**0.942(0.03)**
1-NN	0.832(0.04)	0.839(0.03)	0.828(0.03)	0.825(0.05)	0.804(0.01)
k-NN	0.854(0.1)	0.853(0.06)	0.861(0.07)	0.839(0.06)	0.821(0.07)
SVDD	0.909(0.05)	0.888(0.04)	0.88(0.04)	0.874(0.07)	0.875(0.05)

The bolded value is the best within the group.

Table 3.10: Sparsity of classifiers in the test for Bimodal, CSTH, Iris-setosa and versicolor data sets on the noise level (standard deviation in parentheses)

Bimodal					
classifier	0%	5%	10%	15%	20%
two norm	2(0.0)	2(0.0)	2(0.0)	2(0.0)	2(0.0)
max norm	2(0.0)	2(0.0)	3.3(0.64)	3.8(0.75)	2(0.0)
one norm	2.2(0.4)	2(0.0)	2(0.0)	2(0.0)	2(0.0)
mix norm	2(0.0)	2(0.0)	12.1(0.94)	8(0.63)	2(0.0)
MoG*	4	4	6	3	3
PDE†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
1-NN†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
k-NN†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
SVDD	6.2(0.6)	7.8(0.6)	17.2(0.75)	21.7(1.0)	23.2(1.83)
CSTH					
classifier	0%	5%	10%	15%	20%
two norm	2(0.0)	2(0.0)	2(0.0)	2(0.0)	2.3(0.72)
max norm	2(0.0)	2(0.0)	2(0.0)	2(0.0)	2.6(0.49)
one norm	2(0.0)	2(0.0)	2(0.0)	2(0.0)	2(0.0)
mix norm	2(0.0)	2(0.0)	2.6(0.7)	2(0.0)	2.6(0.49)
MoG*	2	2	2	2	2
PDE†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
1-NN†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
k-NN†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
SVDD	2.1(0.3)	6.9(0.94)	7.3(0.9)	4.4(0.8)	8.5(0.5)
Iris-setosa					
classifier	0%	5%	10%	15%	20%
two norm	1(0.0)	1(0.0)	1(0.0)	1(0.0)	1(0.0)
max norm	1(0.0)	1(0.0)	1(0.0)	1(0.0)	1(0.0)
one norm	1(0.0)	1(0.0)	1(0.0)	1(0.0)	1(0.0)
mix norm	1(0.0)	1(0.0)	1(0.0)	1(0.0)	1(0.0)
MoG*	1	1	1	1	1
PDE†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
1-NN†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
k-NN†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
SVDD	2(0.0)	4(0.89)	11.6(0.49)	12.8(0.74)	16.4(1.02)
Iris-versicolor					
classifier	0%	5%	10%	15%	20%
two norm	2(0.0)	2.6(0.75)	1(0.0)	1(0.0)	1(0.0)
max norm	3.83(0.8)	2.6(0.49)	1.6(0.55)	3(0.89)	3.2(1.6)
one norm	2.2(0.4)	2(0.0)	1.6(0.49)	4.2(1.17)	2.6(0.49)
mix norm	14.6(3.38)	3.6(1.02)	3.2(1.30)	3(1.10)	2.2(0.98)
MoG*	1	2	1	2	2
PDE†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
1-NN†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
k-NN†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
SVDD	3.4(1.2)	7.2(0.4)	11.8(0.75)	11.6(0.49)	12(0.0)

* The number of Gaussian functions is pre-specified beforehand by parameter.
† The model has the non-zero terms as many as all the training points.

Table 3.11: Sparsity of classifiers in the test for Iris-virginica, Liver, WBC and Ionosphere data sets on the noise level (standard deviation in parentheses)

Iris-virginica					
classifier	0%	5%	10%	15%	20%
two norm	1.2(0.4)	2.8(1.17)	1(0.0)	3.4(1.62)	1(0.0)
max norm	2.2(0.4)	1(0.0)	1(0.0)	1(0.0)	1(0.0)
one norm	2(0.63)	2.2(0.89)	2.8(0.37)	2.6(0.49)	2.6(0.49)
mix norm	4.6(0.5)	4(0.49)	1(0.0)	1(0.0)	1(0.0)
MoG*	1	2	2	1	2
PDE†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
1-NN†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
k-NN†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
SVDD	4.8(0.75)	10.2(0.98)	11.4(0.49)	16.2(0.75)	17.4(0.49)
Liver					
classifier	0%	5%	10%	15%	20%
two norm	1.2(0.4)	1.7(0.46)	1(0.0)	1(0.0)	1(0.0)
max norm	1(0.0)	1(0.0)	1(0.0)	2(0.63)	1(0.0)
one norm	1.9(0.3)	2(0.45)	2.7(0.49)	1.2(0.4)	1(0.0)
mix norm	1(0.0)	1(0.0)	4.9(1.04)	1(0.0)	1(0.0)
MoG*	2	1	2	2	1
PDE†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
1-NN†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
k-NN†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
SVDD	83(2.68)	85.5(3.04)	89.3(4.22)	95.6(2.33)	96.3(2.93)
WBC					
classifier	0%	5%	10%	15%	20%
two norm	1(0.0)	1(0.0)	1(0.0)	1(0.0)	1(0.0)
max norm	1(0.0)	1(0.0)	1(0.0)	1(0.0)	1(0.0)
one norm	1.11(0.31)	1(0.0)	1(0.0)	1(0.0)	1(0.0)
mix norm	1.9(0.3)	3.7(2.15)	13.3(0.82)	13.7(5.78)	5(0.77)
MoG*	5	4	6	1	3
PDE†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
1-NN†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
k-NN†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
SVDD	13.5(1.02)	29.4(1.50)	35.5(1.91)	49(1.61)	56.1(2.51)
Ionosphere					
classifier	0%	5%	10%	15%	20%
two norm	2.4(0.66)	4.8(0.98)	2(0.0)	2(0.0)	3.2(0.49)
max norm	3(0.0)	7.2(0.98)	24.6(0.92)	3.5(1.11)	8.3(1.19)
one norm	2.2(0.4)	1(0.0)	4.1(0.7)	2(0.0)	5(0.63)
mix norm	1(0.0)	6.1(0.7)	5.5(1.80)	2.2(0.6)	5.3(1.62)
MoG*	9	1	1	1	1
PDE†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
1-NN†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
k-NN†	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$	$\|N\|$
SVDD	11.7(1.1)	7.7(0.9)	32(1.18)	40.7(1.68)	44.1(1.81)

* The number of Gaussian functions is pre-specified beforehand by parameter.
† The model has the non-zero terms as many as all the training points.

We also examined the computational time of NBCs for training, validation and test. Especially, the training time of NBC relies on the size of training data. In Table 3.12, the results are reported for some data sets: Iris-versicolor, Liver and Ionosphere. Time was measured for the best classifier in the test when noise level 0%, 10% and 20% by summing all the iterations during cross validation. It is observed that all the NBCs require more training time compared to the classifiers in the comparison group because NBC is developed by solving integer programming problem: mix norm ball classifier used more time to find the optimal norm ball covering because the three types of norm balls are added in the problem. It is expected that as noises are more included, the training time would be longer. But, there are some cases in Table 3.12 against our expectations since the training time completely depends on the problem size induced by hyper-parameters related to the generation of norm balls. For each noise level, the parameter combination was different.

However, NBCs surely brought their strength in the time of identifying class labels in taking shorter time compared to the other classifiers. Furthermore, the higher dimension it is, the more time the classifiers in the comparison need. The validation and training time of NBCs does not depend on the dimension. This is why NBCs has high sparsity and less testing burden in measuring only the distance between test objects and the center of the norm ball. MoG is a very sparse model but its classifier spend more time in the validation and test due to the calculation of the inverse of covariance matrix.

Additional experiment was conducted in order to see what would happened for large scale data. Given the parameter combination of Bimodal data show-

Table 3.12: Computational time for Iris-versicolor, Liver, Ionosphere data (in seconds)

Iris-versicolor									
	0%			10%			20%		
classifier	Training	Validation	Test	Training	Validation	Test	Training	Validation	Test
two norm	0.079	*	*	0.047	*	*	0.032	*	0.032
max norm	0.047	*	*	*	*	*	0.016	*	*
one norm	0.078	*	*	0.016	*	*	0.016	*	*
mix norm	0.22	*	*	0.02	*	*	0.018	*	0.001
MoG	0.019	0.016	*	0.016	*	0.016	*	0.016	0.032
PDE	*	0.017	*	*	0.016	0.016	*	0.016	0.016
1-NN	*	0.0625	0.0313	*	0.075	0	*	0.047	0.047
k-NN	*	0.0625	*	*	0.047	0.017	*	0.031	0.031
SVDD	*	*	*	*	*	*	*	*	*
Liver									
	0%			10%			20%		
classifier	Training	Validation	Test	Training	Validation	Test	Training	Validation	Test
two norm	0.94	*	*	3.126	*	*	0.12	*	*
max norm	0.07	*	*	0.09	*	*	0.091	*	*
one norm	0.35	*	*	2.558	*	*	0.062	*	*
mix norm	0.69	*	*	2.884	0.003	0.002	3.95	0.001	*
MoG	*	0.032	0.032	0.048	0.048	0.016	0.047	0.032	0.032
PDE	*	0.016	0.047	*	0.047	0.016	*	0.031	0.031
1-NN	*	0.078	0.047	*	0.125	0.047	*	0.063	0.078
k-NN	*	0.141	*	*	0.094	0.062	*	0.011	0.094
SVDD	0.016	*	0.016	0.016	*	*	0.016	*	*
Ionosphere									
	0%			10%			20%		
classifier	Training	Validation	Test	Training	Validation	Test	Training	Validation	Test
two norm	1.98	0.002	0.003	1.799	*	*	2.29	0.01	0.01
max norm	2.66	0.006	0.01	3.84	0.02	0.02	0.267	*	*
one norm	5.65	*	*	0.547	*	*	2.45	*	*
mix norm	20.98	*	*	0.05	0.005	0.01	0.011	0.016	0.015
MoG	0.011	0.048	0.064	0.016	0.048	0.096	0.048	0.048	0.048
PDE	*	*	*	*	0.063	0.063	*	0.062	0.031
1-NN	*	0.094	0.078	*	0.078	0.094	*	0.094	0.078
k-NN	*	0.094	0.141	*	0.078	0.109	*	0.094	0.078
SVDD	*	*	*	0.016	*	*	0.032	*	0.016

*< 0.0001

Table 3.13: Training time, sparsity and average F_1 score in the test of the NBCs corresponding to parameter combination with the best performance on Bimodal data 100 instances (standard deviations in parentheses)

	100 instances			500 instances			1000 instances		
classifier	time (sec)	sparsity	F_1 score	time (sec)	sparsity	F_1 score	time (sec)	sparsity	F_1 score
two norm	0.141	2(0.0)	0.989(0.03)	11.98	2(0.0)	0.992(0.01)	156.66	2(0.0)	0.996(0.01)
max norm	0.045	2(0.0)	0.965(0.06)	9.59	2(0.0)	0.988(0.01)	129.15	2(0.0)	0.994(0.01)
one norm	0.233	2.2(0.4)	0.96(0.06)	9.08	2(0.0)	0.989(0.01)	122.27	2(0.0)	0.996(0.01)
mix norm	0.2	2(0.0)	0.986(0.03)	17.33	2(0.0)	0.987(0.01)	213.1	2(0.0)	0.994(0.01)

ing the best performance without noises, we made the size of target instances grower. The result in terms of 10-fold cross validation is illustrated in Table 3.13. Despite of the increase of data seize, sparsity and F_1 score remains stable. In contrast, the training time increased by at most 200, 2000 times based on 100 instances even though the size of target data did 5 to 10 times. It demonstrates that the computational time during training increases very rapidly, which leads to critical drawback of NBC.

Chapter 4

Conclusions

4.1 Discussions

We have some criteria for good OCC classifiers: interpretability in the original space, sparsity & testing burden, Sensitivity to noises, the number of hyper-parameters and training/learning process. In terms of interpretability in the orginal space, norm balls are intentionally chosen for data description because their center and radius can be helpful for its purpose.

For sparsity & testing burden, our classifier is sparse and makes less effort on testing objects. The number of norm balls consisting of the classifier is lower than the classifiers in comparison group. It means that NBC do not need to always carry with all training points (as references) when testing such as PDE or Nearest Neighbor Method. Also, compared to the most sparse model, MoG, NBC spend less time in determining the class labels only by computing the distance not the inverse of the covariance matrix. With high sparsity, *edge computing* in mobile or small devices [1, 9] highly makes use of our classifier. Recently, the computational ability of small devices becomes powerful. Owing to accessibility to real-time data and convenience, there has been demands to

analyze the data *i.e.* data mining using their hardware located in the edge of the large network system. If the training is attributed to the main server, and the obtained classifier is installed on the edge devices, then they are able to easily conduct data analytics by testing observations with the small number of norm balls despite of their physical or hardware limitation.

In terms of noises, NBCs shows the comparable results. Especially, maximum norm ball classifiers was robust despite of the increase of noise level. It implies that maximum norm ball classifier can have potential for applications where there is limitation of eliminating noises. It is also very useful for interpretability because it gives the stable performance even though it did not always performed well. On the other hands, mix norm ball classifier has some fluctuations of performance depending on the noise level in high dimensional space. This classifier has many choices to cover the data possibly leading to overfitting. It is easy to adapt to a variety of data. In result, its performance remains unstable when noises exist in training set.

Our classifier requires relatively many hyper-parameters. They are often called free parameters which a user can freely choose. Since there many combinations of them, it could be difficult to narrow down the optimal combinations. However, our model has more learning parameters than SVDD, nearest neighbor method, but it is possible to intuitively estimate them $(\underline{\gamma}, \bar{\gamma}, \mu)$ using the information easily retrieved from the given data or search some values (C) within the specified interval $(\frac{1}{|N|} \leq C)$.

To construct a NBC, we need to find possible norm ball candidates and solve an integer optimization problem, which could be computationally burden.

By applying **Algorithm 1**, the number of candidates norm balls is at most $|N| * (|N| - 1)$, polynomial size of the given training data. The polynomial sized optimization is easily initialized with the candidates from adjustment of hyper-parameters, which makes the warm-start of the problem possible. The problem of obtaining a NBC can be simply modelled, but it takes time to solve it compared to other methods which have efficient commercial algorithms. Also, it is cumbersome that the generation of norm balls candidates is needed as pre-processing.

4.2 Further Research

The selection of norm type are not discussed in detail. In our framework, norm plays a role in kernel trick functions. We do not fully provide the specific instructions or intuition for the choice of norm. Although maximum norm ball shows robust results in noise experiment, it is not certain that maximum norm performs steadily in other data sets. For clarity, additional experiments with a variety of data sets should be scheduled in order to demonstrate the role of norm.

Our model sometimes performed better or comparably. In fact, when radius candidates are created, only the distances between points are chosen. After obtaining the classifier in the test, post preprocessing can be applied such as the change of the radius of the norm balls in the classifier in order to raise its performance by eliminating rigidity of radius.

To build our classifier, the computational time is more needed because of the complexity of P_{NBC}. There is a chance of finding approximation algorithm, efficient ways or dynamic programming by thoroughly examining this problem: for example, the special structure in the problem is recognized. Actually, our problem is similar to set covering problem, so many algorithms can be developed.

Lastly, LP relaxation of P_{NBC} can be considered for the distinctive drawback of our model, training time. If the problem has fractional solutions, then our classifier is the weighted sum of norm balls. For a test object $\mathbf{z}$, we have

$$f(\mathbf{z}) = \sum_{i \in B} w_i I_i(\mathbf{z}) + w_0 \tag{4.1}$$

where w_i is the weight of norm ball i, $I_i(\mathbf{z})$ is the indicator function of ball i and w_0 is the bias term. In order to build the classifier, the following problem should be solved.

If the above concept of OCC works, it could be a more efficient approach in terms of training. Furthermore, It is possible to expand to regression tasks because $f(\mathbf{x}) = \sum_{i \in B} w_i I_i(\mathbf{z}) + w_0$ computes a value for each input $\mathbf{z}$.

Chapter 5

Appendix

Table 5.1: The average recall, precision and F_1-score of Bimodal data depending on the noise level (standard deviation in parentheses)

0%									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.992(0.0)	1.0(0.0)	0.996(0.0)	0.91(0.16)	1.0(0.0)	0.944(0.1)	0.98(0.06)	1.0(0.0)	**0.989(0.03)**
max norm	0.999(0.0)	1.0(0.0)	0.999(0.0)	0.98(0.06)	0.974(0.05)	0.975(0.04)	0.96(0.07)	0.972(0.06)	0.965(0.06)
one norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.97(0.06)	0.963(0.06)	0.965(0.05)	0.96(0.07)	0.961(0.07)	0.96(0.06)
mix norm	0.998(0.0)	1.0(0.0)	0.999(0.0)	0.99(0.03)	0.974(0.05)	0.981(0.03)	1.0(0.0)	0.974(0.05)	0.986(0.03)
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.99(0.03)	0.974(0.05)	0.981(0.03)	0.96(0.09)	0.964(0.06)	0.959(0.06)
PDE	0.98(0.04)	1.0(0.0)	0.989(0.02)	0.98(0.04)	0.973(0.06)	0.976(0.04)	0.98(0.04)	0.973(0.06)	**0.976(0.04)**
1-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.91(0.09)	0.923(0.07)	0.913(0.07)	0.93(0.05)	0.924(0.07)	0.926(0.05)
k-NN	0.998(0.01)	1.0(0.0)	0.999(0.0)	0.98(0.06)	0.965(0.06)	0.97(0.04)	0.95(0.1)	0.911(0.07)	0.925(0.06)
SVDD	0.96(0.02)	1.0(0.0)	0.98(0.01)	0.89(0.14)	0.979(0.04)	0.926(0.09)	0.92(0.07)	0.978(0.04)	0.947(0.06)
5 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.989(0.0)	0.99(0.0)	0.99(0.0)	0.99(0.03)	0.919(0.1)	0.951(0.07)	0.99(0.03)	0.911(0.1)	0.946(0.06)
max norm	1.0(0.0)	0.99(0.0)	0.995(0.0)	0.98(0.06)	0.933(0.09)	0.953(0.06)	0.99(0.03)	0.925(0.08)	0.954(0.05)
one norm	1.0(0.0)	0.99(0.0)	0.995(0.0)	0.99(0.03)	0.91(0.1)	0.946(0.06)	0.99(0.03)	0.911(0.1)	0.946(0.06)
mix norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.927(0.09)	0.96(0.05)	1.0(0.0)	0.927(0.09)	**0.96(0.05)**
MoG	1.0(0.0)	0.996(0.01)	0.998(0.0)	1.0(0.0)	0.918(0.09)	0.955(0.05)	0.98(0.06)	0.906(0.09)	0.94(0.07)
PDE	1.0(0.0)	0.993(0.01)	0.996(0.0)	1.0(0.0)	0.912(0.1)	0.951(0.05)	1.0(0.0)	0.903(0.09)	**0.946(0.05)**
1-NN	1.0(0.0)	0.944(0.01)	0.971(0.01)	0.91(0.09)	0.684(0.12)	0.776(0.09)	0.93(0.05)	0.693(0.11)	0.79(0.08)
k-NN	1.0(0.0)	0.996(0.01)	0.998(0.0)	0.99(0.03)	0.91(0.1)	0.946(0.07)	1.0(0.0)	0.871(0.12)	0.927(0.07)
SVDD	0.971(0.01)	0.963(0.02)	0.967(0.01)	0.95(0.07)	0.883(0.1)	0.912(0.08)	0.95(0.05)	0.875(0.11)	0.908(0.07)
10 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.973(0.01)	0.994(0.01)	0.983(0.01)	0.97(0.05)	0.926(0.07)	0.947(0.05)	0.93(0.09)	0.941(0.08)	**0.934(0.08)**
max norm	1.0(0.0)	0.971(0.01)	0.985(0.0)	1.0(0.0)	0.888(0.1)	0.938(0.05)	0.99(0.03)	0.862(0.09)	0.919(0.05)
one norm	0.982(0.01)	0.998(0.0)	0.989(0.01)	0.96(0.08)	0.934(0.07)	0.946(0.07)	0.92(0.08)	0.931(0.08)	0.925(0.07)
mix norm	0.999(0.0)	0.902(0.01)	0.948(0.0)	1.0(0.0)	0.923(0.07)	0.958(0.04)	0.96(0.08)	0.913(0.08)	0.932(0.06)
MoG	1.0(0.0)	0.993(0.01)	0.996(0.0)	1.0(0.0)	0.907(0.07)	0.95(0.04)	0.99(0.03)	0.897(0.07)	0.94(0.05)
PDE	1.0(0.0)	0.99(0.0)	0.995(0.0)	1.0(0.0)	0.907(0.07)	0.95(0.04)	1.0(0.0)	0.907(0.07)	**0.95(0.04)**
1-NN	1.0(0.0)	0.901(0.0)	0.948(0.0)	0.9(0.09)	0.606(0.09)	0.72(0.07)	0.91(0.07)	0.601(0.09)	0.719(0.07)
k-NN	0.926(0.03)	0.992(0.01)	0.958(0.02)	0.98(0.04)	0.908(0.04)	0.942(0.04)	0.97(0.05)	0.894(0.06)	0.929(0.04)
SVDD	0.946(0.01)	0.951(0.01)	0.949(0.01)	0.91(0.09)	0.829(0.11)	0.864(0.09)	0.85(0.12)	0.81(0.1)	0.828(0.1)

15%									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.993(0.01)	0.966(0.02)	0.979(0.01)	0.96(0.07)	0.942(0.09)	0.948(0.06)	0.96(0.05)	0.925(0.09)	**0.939(0.06)**
max norm	1.0(0.0)	0.935(0.01)	0.966(0.01)	1.0(0.0)	0.888(0.09)	0.938(0.05)	1.0(0.0)	0.863(0.12)	0.923(0.07)
one norm	0.98(0.01)	0.962(0.01)	0.971(0.01)	0.97(0.05)	0.948(0.08)	0.957(0.05)	0.95(0.07)	0.927(0.11)	0.934(0.07)
mix norm	0.995(0.01)	0.965(0.01)	0.98(0.01)	0.97(0.05)	0.948(0.08)	0.957(0.05)	0.94(0.08)	0.942(0.09)	0.936(0.06)
MoG	1.0(0.0)	0.959(0.01)	0.979(0.0)	1.0(0.0)	0.918(0.09)	0.955(0.05)	1.0(0.0)	0.911(0.09)	0.951(0.05)
PDE	1.0(0.0)	0.954(0.01)	0.976(0.0)	1.0(0.0)	0.918(0.09)	0.955(0.05)	1.0(0.0)	0.928(0.09)	**0.96(0.05)**
1-NN	1.0(0.0)	0.848(0.01)	0.917(0.0)	0.9(0.08)	0.562(0.06)	0.691(0.06)	0.88(0.09)	0.554(0.05)	0.679(0.06)
k-NN	0.995(0.01)	0.95(0.01)	0.972(0.0)	1.0(0.0)	0.911(0.09)	0.951(0.05)	0.98(0.06)	0.839(0.08)	0.901(0.06)
SVDD	0.932(0.01)	0.98(0.01)	0.956(0.01)	0.87(0.13)	0.961(0.09)	0.908(0.09)	0.88(0.12)	0.966(0.08)	0.914(0.07)
20 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.987(0.01)	0.943(0.01)	0.964(0.01)	0.99(0.03)	0.932(0.07)	0.958(0.04)	0.98(0.06)	0.932(0.07)	**0.952(0.04)**
max norm	1.0(0.0)	0.917(0.02)	0.956(0.01)	0.99(0.03)	0.888(0.1)	0.933(0.05)	0.98(0.04)	0.871(0.1)	0.919(0.06)
one norm	1.0(0.0)	0.941(0.01)	0.97(0.01)	0.98(0.06)	0.921(0.07)	0.947(0.05)	0.98(0.06)	0.921(0.07)	0.947(0.05)
mix norm	1.0(0.0)	0.944(0.01)	0.971(0.01)	1.0(0.0)	0.909(0.08)	0.95(0.05)	1.0(0.0)	0.909(0.08)	0.95(0.05)
MoG	1.0(0.0)	0.952(0.01)	0.976(0.01)	1.0(0.0)	0.923(0.07)	0.958(0.04)	1.0(0.0)	0.923(0.07)	**0.958(0.04)**
PDE	1.0(0.0)	0.951(0.01)	0.975(0.0)	1.0(0.0)	0.923(0.07)	0.958(0.04)	1.0(0.0)	0.923(0.07)	**0.958(0.04)**
1-NN	1.0(0.0)	0.8(0.01)	0.889(0.0)	0.96(0.07)	0.555(0.04)	0.702(0.04)	0.94(0.05)	0.528(0.05)	0.675(0.05)
k-NN	0.972(0.02)	0.947(0.01)	0.959(0.01)	0.99(0.03)	0.889(0.07)	0.935(0.05)	1.0(0.0)	0.898(0.07)	0.945(0.04)
SVDD	0.942(0.01)	0.86(0.02)	0.899(0.02)	0.88(0.14)	0.881(0.12)	0.869(0.1)	0.85(0.12)	0.832(0.14)	0.826(0.07)

The bolded value is the best within the group.

Table 5.2: The average recall, precision and F_1-score of CSTH data depending on the noise level (standard deviation in parentheses)

0%									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.984(0.01)	1.0(0.0)	0.992(0.0)	0.91(0.16)	1.0(0.0)	0.944(0.1)	0.98(0.06)	1.0(0.0)	0.989(0.03)
max norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.88(0.2)	1.0(0.0)	0.922(0.14)	0.91(0.12)	1.0(0.0)	0.948(0.07)
one norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.88(0.2)	1.0(0.0)	0.922(0.13)	0.99(0.03)	1.0(0.0)	**0.995(0.02)**
mix norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.88(0.2)	1.0(0.0)	0.922(0.14)	0.91(0.12)	1.0(0.0)	0.948(0.07)
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
PDE	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
1-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.92(0.16)	1.0(0.0)	0.95(0.1)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
k-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
SVDD	0.992(0.02)	1.0(0.0)	0.996(0.01)	0.87(0.3)	0.9(0.3)	0.884(0.3)	0.88(0.21)	0.938(0.19)	0.906(0.2)
5 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.985(0.0)	1.0(0.0)	0.993(0.0)	0.86(0.26)	1.0(0.0)	0.895(0.21)	0.9(0.19)	1.0(0.0)	0.934(0.13)
max norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.88(0.2)	1.0(0.0)	0.922(0.14)	0.9(0.13)	1.0(0.0)	0.942(0.07)
one norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.88(0.2)	1.0(0.0)	0.922(0.13)	0.93(0.18)	1.0(0.0)	**0.952(0.13)**
mix norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.88(0.2)	1.0(0.0)	0.922(0.14)	0.9(0.13)	1.0(0.0)	0.942(0.07)
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
PDE	1.0(0.0)	0.944(0.01)	0.971(0.01)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
1-NN	1.0(0.0)	0.944(0.01)	0.971(0.01)	0.89(0.18)	0.885(0.15)	0.869(0.13)	0.99(0.03)	0.874(0.16)	0.922(0.11)
k-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
SVDD	0.948(0.01)	0.956(0.01)	0.952(0.01)	0.95(0.09)	0.973(0.04)	0.958(0.05)	0.93(0.1)	0.972(0.04)	0.946(0.06)
10 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.985(0.0)	1.0(0.0)	0.993(0.0)	0.86(0.26)	1.0(0.0)	0.895(0.21)	0.92(0.12)	1.0(0.0)	0.954(0.07)
max norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.88(0.2)	1.0(0.0)	0.922(0.14)	0.93(0.11)	1.0(0.0)	0.96(0.06)
one norm	1.0(0.0)	0.999(0.0)	0.999(0.0)	0.91(0.16)	0.971(0.09)	0.928(0.11)	0.99(0.03)	0.967(0.1)	**0.975(0.06)**
mix norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.88(0.19)	1.0(0.0)	0.923(0.13)	0.86(0.22)	1.0(0.0)	0.906(0.15)
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
PDE	1.0(0.0)	0.904(0.01)	0.95(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
1-NN	1.0(0.0)	0.901(0.0)	0.948(0.0)	0.92(0.16)	0.822(0.15)	0.851(0.13)	1.0(0.0)	0.833(0.15)	0.9(0.11)
k-NN	1.0(0.0)	0.999(0.0)	0.999(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.991(0.03)	0.995(0.01)
SVDD	0.962(0.02)	0.918(0.01)	0.939(0.01)	0.94(0.09)	0.884(0.12)	0.905(0.1)	0.94(0.1)	0.879(0.13)	0.899(0.09)

15%									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.985(0.0)	1.0(0.0)	0.993(0.0)	0.86(0.26)	1.0(0.0)	0.895(0.21)	0.92(0.12)	1.0(0.0)	0.954(0.07)
max norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.88(0.2)	1.0(0.0)	0.922(0.14)	0.93(0.11)	1.0(0.0)	**0.96(0.06)**
one norm	1.0(0.0)	0.999(0.0)	0.999(0.0)	0.91(0.16)	1.0(0.0)	0.944(0.1)	0.99(0.03)	0.967(0.1)	0.975(0.06)
mix norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.88(0.2)	1.0(0.0)	0.922(0.14)	0.93(0.11)	1.0(0.0)	0.96(0.06)
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
PDE	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
1-NN	1.0(0.0)	0.848(0.01)	0.917(0.0)	0.92(0.16)	0.828(0.04)	0.865(0.09)	1.0(0.0)	0.842(0.04)	0.914(0.02)
k-NN	0.995(0.01)	0.995(0.01)	0.995(0.01)	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.98(0.04)	1.0(0.0)	0.989(0.02)
SVDD	0.984(0.02)	0.854(0.01)	0.914(0.01)	0.94(0.09)	0.799(0.12)	0.857(0.1)	0.96(0.07)	0.823(0.12)	0.88(0.08)
20 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.985(0.01)	0.989(0.03)	0.987(0.02)	0.84(0.27)	1.0(0.0)	0.882(0.21)	0.92(0.12)	1.0(0.0)	0.954(0.07)
max norm	1.0(0.0)	0.925(0.06)	0.96(0.03)	0.88(0.2)	0.991(0.03)	0.917(0.13)	0.93(0.11)	0.971(0.06)	0.946(0.07)
one norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.86(0.2)	1.0(0.0)	0.911(0.13)	0.99(0.03)	1.0(0.0)	**0.995(0.02)**
mix norm	1.0(0.0)	0.925(0.06)	0.96(0.03)	0.88(0.2)	0.991(0.03)	0.917(0.13)	0.93(0.11)	0.971(0.06)	0.946(0.07)
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
PDE	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
1-NN	1.0(0.0)	0.8(0.01)	0.889(0.0)	0.92(0.16)	0.768(0.07)	0.831(0.1)	1.0(0.0)	0.814(0.03)	0.897(0.02)
k-NN	1.0(0.0)	0.926(0.06)	0.96(0.03)	1.0(0.0)	0.991(0.03)	0.995(0.01)	0.97(0.06)	0.954(0.07)	0.961(0.06)
SVDD	0.959(0.01)	0.815(0.01)	0.881(0.01)	0.94(0.09)	0.82(0.11)	0.87(0.08)	0.94(0.1)	0.812(0.11)	0.865(0.09)

The bolded value is the best within the group.

Table 5.3: The average recall, precision and F_1-score of Iris-setosa data depending on the noise level (standard deviation in parentheses)

0%									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.987(0.02)	1.0(0.0)	0.993(0.01)	0.98(0.04)	1.0(0.0)	0.989(0.02)	0.98(0.04)	1.0(0.0)	0.989(0.02)
max norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.98(0.04)	1.0(0.0)	0.989(0.02)	0.98(0.04)	1.0(0.0)	0.989(0.02)
one norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.92(0.12)	1.0(0.0)	0.954(0.07)
mix norm	0.993(0.01)	1.0(0.0)	0.997(0.01)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
PDE	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
1-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
k-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
SVDD	0.987(0.03)	1.0(0.0)	0.993(0.01)	0.94(0.08)	1.0(0.0)	0.967(0.04)	0.94(0.08)	1.0(0.0)	0.967(0.04)
5 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.974(0.01)	1.0(0.0)	0.987(0.01)	0.98(0.04)	1.0(0.0)	0.989(0.02)	0.94(0.08)	1.0(0.0)	0.967(0.04)
max norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.98(0.04)	1.0(0.0)	0.989(0.02)	0.96(0.05)	1.0(0.0)	0.979(0.03)
one norm	0.993(0.01)	1.0(0.0)	0.997(0.01)	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.92(0.12)	1.0(0.0)	0.954(0.07)
mix norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.98(0.04)	1.0(0.0)	0.989(0.02)	0.98(0.04)	1.0(0.0)	**0.989(0.02)**
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
PDE	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
1-NN	1.0(0.0)	0.944(0.02)	0.971(0.01)	0.86(0.1)	0.467(0.17)	0.58(0.12)	0.86(0.08)	0.402(0.08)	0.54(0.06)
k-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
SVDD	0.953(0.02)	0.954(0.03)	0.954(0.02)	0.92(0.07)	0.98(0.04)	0.947(0.05)	0.94(0.08)	0.982(0.04)	0.958(0.04)
10 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.974(0.01)	1.0(0.0)	0.987(0.01)	0.98(0.04)	1.0(0.0)	0.989(0.02)	0.94(0.08)	1.0(0.0)	0.967(0.04)
max norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.98(0.04)	1.0(0.0)	0.989(0.02)	0.96(0.05)	1.0(0.0)	0.979(0.03)
one norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.98(0.04)	1.0(0.0)	**0.989(0.02)**
mix norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.98(0.04)	1.0(0.0)	0.989(0.02)	0.96(0.05)	1.0(0.0)	0.979(0.03)
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
PDE	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
1-NN	1.0(0.0)	0.893(0.01)	0.943(0.01)	0.86(0.1)	0.439(0.11)	0.573(0.1)	0.86(0.08)	0.426(0.11)	0.558(0.08)
k-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
SVDD	0.8(0.02)	1.0(0.0)	0.889(0.01)	0.84(0.12)	1.0(0.0)	0.908(0.08)	0.78(0.1)	1.0(0.0)	0.873(0.07)

15%									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.974(0.01)	1.0(0.0)	0.987(0.01)	0.98(0.04)	1.0(0.0)	0.989(0.02)	0.94(0.08)	1.0(0.0)	0.967(0.04)
max norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.98(0.04)	1.0(0.0)	0.989(0.02)	0.96(0.05)	1.0(0.0)	0.979(0.03)
one norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.98(0.04)	1.0(0.0)	0.989(0.02)	0.94(0.08)	1.0(0.0)	0.967(0.04)
mix norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.98(0.04)	1.0(0.0)	0.989(0.02)	0.98(0.04)	1.0(0.0)	**0.989(0.02)**
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
PDE	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
1-NN	1.0(0.0)	0.848(0.01)	0.918(0.01)	0.98(0.04)	0.379(0.06)	0.544(0.07)	0.98(0.04)	0.371(0.06)	0.536(0.07)
k-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
SVDD	0.827(0.02)	1.0(0.0)	0.905(0.01)	0.84(0.14)	0.905(0.13)	0.866(0.12)	0.8(0.11)	0.898(0.16)	0.83(0.08)
20 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.974(0.01)	1.0(0.0)	0.987(0.01)	0.98(0.04)	1.0(0.0)	0.989(0.02)	0.94(0.08)	1.0(0.0)	0.967(0.04)
max norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.98(0.04)	1.0(0.0)	0.989(0.02)	0.96(0.05)	1.0(0.0)	0.979(0.03)
one norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.98(0.04)	1.0(0.0)	**0.989(0.02)**
mix norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.98(0.04)	1.0(0.0)	0.989(0.02)	0.98(0.04)	1.0(0.0)	**0.989(0.02)**
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
PDE	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
1-NN	1.0(0.0)	0.794(0.02)	0.885(0.01)	0.76(0.14)	0.518(0.17)	0.609(0.17)	0.8(0.06)	0.382(0.1)	0.51(0.08)
k-NN	0.973(0.01)	1.0(0.0)	0.986(0.01)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	**1.0(0.0)**
SVDD	0.76(0.03)	0.991(0.02)	0.86(0.02)	0.74(0.14)	0.917(0.17)	0.806(0.13)	0.74(0.14)	0.94(0.12)	0.818(0.11)

The bolded value is the best within the group.

Table 5.4: The average recall, precision and F_1-score of Iris-versicolor data depending on the noise level (standard deviation in parentheses)

0%									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.987(0.02)	1.0(0.0)	0.993(0.01)	0.9(0.09)	0.934(0.09)	0.909(0.05)	0.88(0.12)	0.828(0.17)	0.837(0.1)
max norm	0.993(0.01)	1.0(0.0)	0.997(0.01)	0.88(0.08)	0.905(0.12)	0.883(0.05)	0.84(0.08)	0.859(0.15)	0.845(0.11)
one norm	0.973(0.02)	1.0(0.0)	0.986(0.01)	0.98(0.04)	0.917(0.08)	0.944(0.04)	0.9(0.11)	0.851(0.12)	**0.867(0.09)**
mix norm	0.993(0.01)	1.0(0.0)	0.997(0.01)	1.0(0.0)	0.781(0.13)	0.871(0.09)	0.96(0.05)	0.752(0.21)	0.83(0.16)
MoG	0.993(0.01)	1.0(0.0)	0.997(0.01)	0.98(0.04)	0.925(0.11)	0.947(0.06)	0.96(0.05)	0.863(0.13)	0.903(0.08)
PDE	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.902(0.09)	0.946(0.05)	0.98(0.04)	0.854(0.15)	**0.907(0.1)**
1-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.86(0.1)	0.83(0.13)	0.83(0.04)	0.9(0.15)	0.772(0.2)	0.796(0.11)
k-NN	0.973(0.02)	1.0(0.0)	0.986(0.01)	0.96(0.05)	0.838(0.13)	0.887(0.06)	0.96(0.05)	0.799(0.17)	0.863(0.11)
SVDD	0.94(0.04)	1.0(0.0)	0.969(0.02)	0.9(0.11)	0.892(0.1)	0.891(0.08)	0.86(0.1)	0.9(0.13)	0.871(0.08)
5 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.98(0.02)	1.0(0.0)	0.99(0.01)	0.96(0.08)	0.851(0.18)	0.886(0.1)	0.96(0.08)	0.838(0.16)	0.883(0.1)
max norm	0.947(0.04)	0.98(0.04)	0.963(0.04)	0.94(0.08)	0.831(0.17)	0.874(0.12)	0.9(0.11)	0.86(0.17)	0.873(0.14)
one norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.906(0.1)	0.948(0.06)	0.98(0.04)	0.879(0.15)	**0.919(0.1)**
mix norm	0.947(0.05)	0.979(0.04)	0.962(0.04)	0.9(0.06)	0.893(0.12)	0.895(0.09)	0.82(0.12)	0.8(0.23)	0.791(0.16)
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.98(0.04)	0.892(0.15)	0.925(0.09)	1.0(0.0)	0.818(0.17)	**0.89(0.1)**
PDE	0.92(0.07)	1.0(0.0)	0.957(0.04)	0.92(0.07)	0.876(0.16)	0.893(0.12)	0.92(0.07)	0.861(0.16)	0.885(0.11)
1-NN	1.0(0.0)	0.944(0.02)	0.971(0.01)	0.86(0.1)	0.472(0.2)	0.593(0.18)	0.9(0.15)	0.392(0.1)	0.544(0.12)
k-NN	0.973(0.02)	1.0(0.0)	0.986(0.01)	0.96(0.05)	0.845(0.17)	0.886(0.1)	0.96(0.05)	0.801(0.2)	0.862(0.14)
SVDD	0.853(0.02)	1.0(0.0)	0.921(0.01)	0.86(0.08)	0.954(0.09)	0.897(0.03)	0.82(0.15)	0.967(0.07)	0.875(0.08)
10 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.967(0.02)	1.0(0.0)	0.983(0.01)	0.96(0.08)	0.818(0.15)	0.873(0.09)	0.96(0.08)	0.818(0.15)	**0.873(0.09)**
max norm	0.967(0.04)	1.0(0.0)	0.983(0.02)	0.9(0.11)	0.853(0.18)	0.858(0.1)	0.92(0.1)	0.836(0.16)	0.868(0.11)
one norm	0.92(0.02)	1.0(0.0)	0.958(0.01)	0.9(0.11)	0.867(0.14)	0.87(0.08)	0.84(0.08)	0.888(0.12)	0.86(0.09)
mix norm	0.833(0.1)	1.0(0.0)	0.906(0.06)	0.8(0.14)	0.933(0.09)	0.855(0.1)	0.64(0.27)	0.894(0.21)	0.683(0.22)
MoG	0.993(0.01)	1.0(0.0)	0.997(0.01)	0.98(0.04)	0.9(0.13)	0.931(0.07)	0.96(0.05)	0.849(0.14)	**0.895(0.08)**
PDE	0.92(0.07)	1.0(0.0)	0.957(0.04)	0.92(0.07)	0.885(0.14)	0.899(0.11)	0.9(0.09)	0.859(0.16)	0.873(0.12)
1-NN	1.0(0.0)	0.893(0.01)	0.943(0.01)	0.86(0.1)	0.469(0.17)	0.595(0.16)	0.9(0.15)	0.405(0.08)	0.558(0.1)
k-NN	0.973(0.02)	1.0(0.0)	0.986(0.01)	0.96(0.05)	0.833(0.14)	0.882(0.08)	0.96(0.05)	0.76(0.16)	0.841(0.11)
SVDD	0.773(0.01)	1.0(0.0)	0.872(0.01)	0.8(0.13)	1.0(0.0)	0.883(0.08)	0.76(0.15)	0.982(0.04)	0.846(0.08)

15%									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.927(0.02)	0.993(0.01)	0.959(0.01)	0.92(0.08)	0.833(0.21)	0.862(0.14)	0.88(0.08)	0.792(0.22)	**0.813(0.12)**
max norm	0.907(0.08)	0.954(0.04)	0.927(0.04)	0.92(0.08)	0.832(0.18)	0.86(0.11)	0.86(0.14)	0.788(0.18)	0.809(0.12)
one norm	0.907(0.06)	0.964(0.05)	0.934(0.05)	0.86(0.08)	0.96(0.08)	0.905(0.07)	0.76(0.08)	0.879(0.15)	0.812(0.1)
mix norm	0.907(0.06)	0.974(0.04)	0.937(0.04)	0.88(0.08)	0.836(0.11)	0.856(0.08)	0.78(0.12)	0.789(0.27)	0.757(0.17)
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.94(0.08)	0.954(0.09)	0.941(0.05)	0.9(0.09)	0.881(0.15)	**0.879(0.07)**
PDE	0.92(0.07)	1.0(0.0)	0.957(0.04)	0.92(0.07)	0.877(0.16)	0.892(0.11)	0.9(0.09)	0.856(0.18)	0.866(0.11)
1-NN	1.0(0.0)	0.848(0.01)	0.918(0.01)	1.0(0.0)	0.338(0.01)	0.505(0.01)	1.0(0.0)	0.336(0.0)	0.503(0.01)
k-NN	0.86(0.08)	1.0(0.0)	0.923(0.05)	0.9(0.06)	0.872(0.16)	0.873(0.07)	0.8(0.11)	0.823(0.18)	0.791(0.07)
SVDD	0.827(0.02)	1.0(0.0)	0.905(0.01)	0.84(0.15)	0.954(0.09)	0.879(0.08)	0.84(0.15)	0.915(0.13)	0.856(0.07)
20 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.927(0.02)	0.973(0.02)	0.949(0.01)	0.9(0.06)	0.825(0.18)	0.854(0.13)	0.88(0.08)	0.737(0.24)	0.777(0.15)
max norm	0.953(0.05)	0.902(0.06)	0.926(0.05)	0.94(0.08)	0.815(0.14)	0.864(0.09)	0.92(0.12)	0.715(0.14)	**0.799(0.12)**
one norm	0.927(0.05)	0.954(0.06)	0.939(0.05)	0.82(0.08)	0.956(0.05)	0.881(0.06)	0.76(0.15)	0.836(0.19)	0.773(0.13)
mix norm	0.887(0.13)	0.934(0.09)	0.908(0.11)	0.9(0.11)	0.837(0.17)	0.849(0.11)	0.78(0.22)	0.793(0.22)	0.756(0.2)
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.94(0.08)	0.948(0.07)	0.94(0.04)	0.9(0.09)	0.875(0.12)	**0.876(0.05)**
PDE	0.88(0.07)	1.0(0.0)	0.935(0.04)	0.88(0.07)	0.861(0.13)	0.867(0.1)	0.9(0.09)	0.844(0.15)	0.864(0.11)
1-NN	1.0(0.0)	0.794(0.02)	0.885(0.01)	0.76(0.19)	0.412(0.18)	0.525(0.18)	0.76(0.22)	0.35(0.12)	0.472(0.13)
k-NN	0.907(0.02)	1.0(0.0)	0.951(0.01)	0.94(0.05)	0.818(0.15)	0.865(0.09)	0.92(0.07)	0.802(0.17)	0.847(0.11)
SVDD	0.88(0.03)	1.0(0.0)	0.936(0.02)	0.92(0.12)	0.936(0.09)	0.919(0.06)	0.84(0.08)	0.885(0.1)	0.858(0.07)

The bolded value is the best within the group.

Table 5.5: The average recall, precision and F_1-score of Iris-virginica data depending on the noise level (standard deviation in parentheses)

0%									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.918(0.03)	1.0(0.0)	0.957(0.02)	0.9(0.09)	0.884(0.11)	0.882(0.04)	0.84(0.15)	0.808(0.16)	0.799(0.06)
max norm	0.953(0.02)	1.0(0.0)	0.976(0.01)	0.9(0.09)	0.922(0.07)	0.907(0.06)	0.86(0.15)	0.871(0.12)	**0.846(0.06)**
one norm	0.973(0.03)	1.0(0.0)	0.986(0.01)	0.9(0.13)	0.875(0.2)	0.861(0.12)	0.838(0.08)	0.866(0.13)	0.84(0.04)
mix norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.738(0.26)	0.822(0.19)	0.98(0.04)	0.739(0.32)	0.795(0.23)
MoG	0.973(0.03)	1.0(0.0)	0.986(0.01)	0.938(0.05)	0.874(0.16)	0.899(0.1)	0.94(0.05)	0.849(0.19)	0.879(0.11)
PDE	0.96(0.08)	1.0(0.0)	0.978(0.04)	0.96(0.08)	0.864(0.13)	0.9(0.06)	0.94(0.08)	0.869(0.14)	**0.892(0.07)**
1-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.9(0.15)	0.706(0.24)	0.75(0.13)	0.8(0.18)	0.777(0.2)	0.76(0.13)
k-NN	0.871(0.04)	1.0(0.0)	0.931(0.02)	0.86(0.08)	0.812(0.16)	0.822(0.07)	0.82(0.1)	0.752(0.21)	0.761(0.1)
SVDD	0.912(0.02)	1.0(0.0)	0.954(0.01)	0.86(0.1)	0.96(0.05)	0.901(0.05)	0.84(0.14)	0.927(0.09)	0.869(0.07)
5 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.925(0.03)	1.0(0.0)	0.961(0.01)	0.86(0.08)	0.84(0.2)	0.827(0.08)	0.82(0.17)	0.761(0.23)	0.741(0.13)
max norm	0.96(0.01)	1.0(0.0)	0.979(0.01)	0.94(0.05)	0.793(0.18)	0.848(0.11)	0.98(0.04)	0.786(0.16)	**0.861(0.1)**
one norm	0.972(0.03)	0.969(0.04)	0.969(0.02)	0.84(0.1)	0.88(0.12)	0.846(0.03)	0.86(0.14)	0.739(0.2)	0.758(0.08)
mix norm	0.966(0.02)	0.974(0.03)	0.969(0.02)	0.92(0.08)	0.836(0.17)	0.859(0.07)	0.9(0.06)	0.746(0.19)	0.798(0.11)
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.94(0.05)	0.876(0.17)	0.894(0.1)	0.9(0.09)	0.814(0.19)	**0.841(0.12)**
PDE	0.78(0.12)	1.0(0.0)	0.872(0.07)	0.78(0.12)	0.939(0.08)	0.843(0.05)	0.78(0.13)	0.87(0.12)	0.807(0.06)
1-NN	1.0(0.0)	0.943(0.02)	0.971(0.01)	0.776(0.15)	0.454(0.2)	0.555(0.18)	0.676(0.14)	0.485(0.21)	0.544(0.17)
k-NN	0.918(0.03)	1.0(0.0)	0.957(0.01)	0.9(0.06)	0.807(0.2)	0.836(0.12)	0.86(0.14)	0.802(0.17)	0.806(0.08)
SVDD	0.755(0.03)	1.0(0.0)	0.86(0.02)	0.76(0.16)	0.95(0.1)	0.826(0.07)	0.74(0.19)	0.902(0.13)	0.785(0.08)
10 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.885(0.03)	1.0(0.0)	0.938(0.02)	0.9(0.09)	0.878(0.15)	0.876(0.08)	0.84(0.15)	0.794(0.16)	0.791(0.07)
max norm	0.973(0.01)	1.0(0.0)	0.986(0.01)	0.94(0.05)	0.776(0.17)	0.839(0.11)	0.98(0.04)	0.77(0.18)	**0.849(0.11)**
one norm	0.979(0.03)	0.894(0.02)	0.935(0.02)	0.733(0.19)	0.511(0.14)	0.583(0.13)	0.769(0.21)	0.444(0.17)	0.553(0.17)
mix norm	0.973(0.01)	1.0(0.0)	0.986(0.01)	0.94(0.05)	0.776(0.17)	0.839(0.11)	0.98(0.04)	0.77(0.18)	**0.849(0.11)**
MoG	0.993(0.01)	1.0(0.0)	0.996(0.01)	0.878(0.07)	0.903(0.1)	0.886(0.07)	0.86(0.1)	0.87(0.12)	**0.859(0.08)**
PDE	0.74(0.15)	1.0(0.0)	0.842(0.09)	0.74(0.15)	0.935(0.08)	0.814(0.08)	0.76(0.14)	0.879(0.11)	0.803(0.08)
1-NN	1.0(0.0)	0.891(0.01)	0.942(0.01)	0.776(0.15)	0.492(0.15)	0.589(0.13)	0.676(0.14)	0.448(0.17)	0.522(0.14)
k-NN	0.918(0.03)	1.0(0.0)	0.957(0.01)	0.9(0.06)	0.779(0.19)	0.821(0.12)	0.86(0.14)	0.774(0.1)	0.8(0.06)
SVDD	0.769(0.03)	1.0(0.0)	0.869(0.02)	0.78(0.15)	0.831(0.14)	0.783(0.06)	0.76(0.19)	0.756(0.2)	0.711(0.03)

15%

	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.823(0.03)	0.961(0.04)	0.887(0.03)	0.8(0.11)	0.787(0.14)	0.784(0.09)	0.74(0.22)	0.731(0.09)	0.723(0.14)
max norm	0.973(0.01)	0.96(0.02)	0.966(0.01)	0.96(0.05)	0.805(0.16)	0.864(0.09)	0.98(0.04)	0.751(0.17)	**0.838(0.1)**
one norm	0.972(0.03)	0.856(0.01)	0.91(0.02)	0.833(0.09)	0.431(0.11)	0.561(0.11)	0.856(0.15)	0.428(0.1)	0.568(0.11)
mix norm	0.973(0.01)	0.96(0.02)	0.966(0.01)	0.96(0.05)	0.805(0.16)	0.864(0.09)	0.98(0.04)	0.751(0.17)	**0.838(0.1)**
MoG	0.952(0.02)	0.972(0.01)	0.962(0.01)	0.858(0.08)	0.901(0.1)	0.875(0.07)	0.838(0.1)	0.851(0.14)	**0.839(0.1)**
PDE	0.72(0.17)	0.976(0.02)	0.816(0.11)	0.72(0.17)	0.876(0.12)	0.772(0.09)	0.74(0.15)	0.843(0.1)	0.771(0.05)
1-NN	1.0(0.0)	0.845(0.01)	0.916(0.01)	0.98(0.04)	0.336(0.02)	0.501(0.03)	0.96(0.05)	0.332(0.03)	0.493(0.04)
k-NN	0.804(0.08)	0.958(0.03)	0.873(0.06)	0.86(0.1)	0.789(0.13)	0.81(0.06)	0.78(0.12)	0.757(0.16)	0.751(0.07)
SVDD	0.734(0.04)	1.0(0.0)	0.846(0.03)	0.656(0.1)	0.95(0.06)	0.768(0.06)	0.613(0.11)	0.938(0.08)	0.736(0.09)

20 %

	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.851(0.03)	0.947(0.02)	0.896(0.02)	0.88(0.08)	0.757(0.21)	0.786(0.1)	0.8(0.13)	0.694(0.26)	0.701(0.12)
max norm	0.932(0.03)	0.958(0.03)	0.945(0.03)	0.9(0.06)	0.817(0.15)	0.844(0.06)	0.9(0.09)	0.767(0.2)	**0.801(0.09)**
one norm	1.0(0.0)	0.795(0.02)	0.886(0.01)	0.9(0.2)	0.389(0.04)	0.529(0.03)	0.86(0.15)	0.354(0.04)	0.499(0.06)
mix norm	0.932(0.03)	0.958(0.03)	0.945(0.03)	0.9(0.06)	0.817(0.15)	0.844(0.06)	0.9(0.09)	0.767(0.2)	**0.801(0.09)**
MoG	1.0(0.0)	0.961(0.02)	0.98(0.01)	0.88(0.1)	0.92(0.16)	0.883(0.08)	0.84(0.14)	0.848(0.19)	**0.835(0.14)**
PDE	0.72(0.19)	0.961(0.02)	0.809(0.13)	0.72(0.19)	0.938(0.12)	0.786(0.1)	0.7(0.18)	0.871(0.16)	0.752(0.1)
1-NN	1.0(0.0)	0.791(0.01)	0.883(0.01)	0.88(0.19)	0.375(0.12)	0.52(0.14)	0.78(0.17)	0.352(0.1)	0.477(0.11)
k-NN	0.824(0.04)	0.976(0.02)	0.892(0.02)	0.86(0.1)	0.74(0.19)	0.77(0.08)	0.798(0.06)	0.761(0.22)	0.754(0.11)
SVDD	0.687(0.03)	0.953(0.0)	0.798(0.02)	0.656(0.1)	0.865(0.12)	0.734(0.06)	0.613(0.11)	0.84(0.2)	0.691(0.1)

The bolded value is the best within the group.

Table 5.6: The average recall, precision and F_1-score of Liver data depending on the noise level (standard deviation in parentheses)

0%									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.908(0.09)	1.0(0.0)	0.95(0.05)	0.929(0.08)	0.445(0.05)	0.601(0.05)	0.937(0.06)	0.439(0.04)	**0.596(0.04)**
max norm	0.984(0.01)	1.0(0.0)	0.992(0.0)	0.979(0.03)	0.425(0.01)	0.593(0.02)	0.972(0.05)	0.425(0.02)	0.591(0.02)
one norm	0.877(0.02)	1.0(0.0)	0.934(0.01)	0.837(0.16)	0.47(0.05)	0.597(0.07)	0.823(0.18)	0.464(0.05)	0.587(0.08)
mix norm	0.984(0.01)	1.0(0.0)	0.992(0.0)	0.986(0.03)	0.427(0.01)	0.596(0.01)	0.986(0.03)	0.427(0.01)	**0.596(0.01)**
MoG	0.938(0.01)	1.0(0.0)	0.968(0.01)	0.867(0.12)	0.448(0.04)	0.589(0.05)	0.86(0.12)	0.448(0.04)	0.588(0.06)
PDE	0.81(0.16)	1.0(0.0)	0.885(0.11)	0.81(0.16)	0.435(0.04)	0.563(0.07)	0.802(0.16)	0.432(0.04)	0.559(0.07)
1-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.958(0.05)	0.428(0.03)	0.591(0.03)	0.951(0.04)	0.428(0.02)	0.59(0.02)
k-NN	0.998(0.0)	1.0(0.0)	0.999(0.0)	0.993(0.02)	0.429(0.02)	0.599(0.02)	0.993(0.02)	0.428(0.02)	**0.598(0.02)**
SVDD	0.427(0.01)	1.0(0.0)	0.598(0.01)	0.274(0.16)	0.534(0.14)	0.346(0.17)	0.289(0.17)	0.557(0.19)	0.364(0.19)
5 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.993(0.0)	0.947(0.01)	0.97(0.0)	0.993(0.02)	0.428(0.01)	0.598(0.02)	0.986(0.03)	0.421(0.01)	0.59(0.01)
max norm	0.986(0.0)	0.946(0.0)	0.965(0.0)	0.986(0.03)	0.426(0.01)	0.594(0.02)	0.986(0.03)	0.425(0.01)	**0.593(0.02)**
one norm	0.998(0.0)	0.951(0.0)	0.974(0.0)	0.986(0.03)	0.43(0.01)	0.599(0.01)	0.972(0.05)	0.425(0.02)	0.591(0.02)
mix norm	0.993(0.01)	0.946(0.0)	0.969(0.0)	1.0(0.0)	0.424(0.01)	0.596(0.01)	0.986(0.03)	0.422(0.01)	0.591(0.02)
MoG	0.905(0.01)	0.955(0.01)	0.929(0.01)	0.887(0.13)	0.44(0.04)	0.586(0.06)	0.866(0.13)	0.426(0.04)	0.57(0.06)
PDE	0.809(0.18)	0.947(0.01)	0.861(0.12)	0.809(0.18)	0.435(0.05)	0.563(0.08)	0.809(0.16)	0.439(0.04)	0.566(0.07)
1-NN	1.0(0.0)	0.947(0.01)	0.973(0.0)	0.979(0.03)	0.423(0.02)	0.59(0.02)	0.972(0.03)	0.422(0.02)	0.589(0.02)
k-NN	0.998(0.0)	0.951(0.01)	0.974(0.0)	0.993(0.02)	0.429(0.02)	0.599(0.02)	0.993(0.02)	0.425(0.02)	**0.595(0.02)**
SVDD	0.628(0.03)	0.956(0.02)	0.757(0.02)	0.281(0.14)	0.535(0.18)	0.34(0.13)	0.296(0.15)	0.571(0.21)	0.364(0.15)
10 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.986(0.0)	0.9(0.0)	0.941(0.0)	0.986(0.03)	0.428(0.01)	0.597(0.01)	0.986(0.03)	0.428(0.01)	**0.597(0.01)**
max norm	0.983(0.01)	0.903(0.01)	0.941(0.0)	0.979(0.03)	0.425(0.01)	0.593(0.01)	0.972(0.05)	0.423(0.02)	0.589(0.02)
one norm	1.0(0.0)	0.9(0.0)	0.948(0.0)	1.0(0.0)	0.428(0.01)	0.6(0.01)	0.986(0.03)	0.418(0.01)	0.587(0.01)
mix norm	0.994(0.01)	0.904(0.01)	0.947(0.01)	0.964(0.06)	0.434(0.02)	0.598(0.03)	0.944(0.07)	0.414(0.02)	0.575(0.03)
MoG	0.938(0.01)	0.916(0.01)	0.927(0.01)	0.867(0.12)	0.445(0.02)	0.586(0.04)	0.86(0.12)	0.441(0.03)	0.582(0.05)
PDE	0.746(0.16)	0.899(0.01)	0.806(0.1)	0.746(0.16)	0.464(0.03)	0.565(0.06)	0.753(0.16)	0.471(0.03)	0.572(0.06)
1-NN	1.0(0.0)	0.899(0.01)	0.947(0.0)	0.972(0.03)	0.425(0.02)	0.591(0.02)	0.965(0.05)	0.422(0.02)	0.587(0.03)
k-NN	0.996(0.0)	0.904(0.01)	0.948(0.0)	0.993(0.02)	0.429(0.01)	0.599(0.02)	0.979(0.05)	0.426(0.01)	**0.594(0.02)**
SVDD	0.633(0.02)	0.91(0.02)	0.747(0.02)	0.281(0.14)	0.536(0.2)	0.341(0.13)	0.296(0.15)	0.526(0.15)	0.363(0.15)

15%									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.968(0.02)	0.846(0.0)	0.903(0.01)	0.979(0.05)	0.432(0.02)	0.599(0.02)	0.965(0.05)	0.424(0.01)	0.589(0.02)
max norm	0.997(0.0)	0.848(0.01)	0.917(0.0)	0.986(0.03)	0.427(0.01)	0.596(0.01)	0.965(0.07)	0.415(0.02)	0.58(0.03)
one norm	0.934(0.07)	0.842(0.01)	0.884(0.04)	0.957(0.06)	0.442(0.05)	0.603(0.05)	0.958(0.05)	0.431(0.02)	**0.594(0.03)**
mix norm	0.986(0.0)	0.843(0.0)	0.909(0.0)	0.986(0.03)	0.425(0.01)	0.594(0.01)	0.986(0.03)	0.425(0.01)	**0.594(0.01)**
MoG	0.938(0.02)	0.863(0.01)	0.899(0.01)	0.874(0.11)	0.44(0.03)	0.585(0.05)	0.86(0.12)	0.44(0.04)	0.582(0.06)
PDE	0.767(0.17)	0.845(0.01)	0.793(0.1)	0.767(0.17)	0.449(0.06)	0.562(0.09)	0.774(0.18)	0.446(0.06)	0.562(0.09)
1-NN	1.0(0.0)	0.845(0.01)	0.916(0.0)	0.986(0.03)	0.43(0.02)	0.599(0.02)	0.986(0.03)	0.427(0.02)	**0.596(0.02)**
k-NN	0.996(0.0)	0.847(0.0)	0.915(0.0)	1.0(0.0)	0.427(0.01)	0.598(0.01)	0.979(0.05)	0.42(0.01)	0.587(0.01)
SVDD	0.681(0.04)	0.86(0.03)	0.759(0.03)	0.296(0.15)	0.504(0.18)	0.362(0.17)	0.303(0.17)	0.535(0.2)	0.375(0.18)
20 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.966(0.03)	0.798(0.0)	0.874(0.01)	0.979(0.05)	0.43(0.01)	0.597(0.02)	0.972(0.05)	0.426(0.01)	0.592(0.02)
max norm	0.986(0.0)	0.796(0.0)	0.88(0.0)	0.986(0.03)	0.425(0.01)	0.594(0.02)	0.986(0.03)	0.424(0.02)	0.593(0.02)
one norm	0.922(0.11)	0.794(0.01)	0.85(0.06)	0.957(0.06)	0.456(0.08)	0.613(0.07)	0.944(0.07)	0.438(0.04)	**0.597(0.04)**
mix norm	0.986(0.0)	0.796(0.0)	0.88(0.0)	0.986(0.03)	0.425(0.01)	0.594(0.02)	0.986(0.03)	0.424(0.02)	0.593(0.02)
MoG	0.905(0.01)	0.8(0.01)	0.849(0.01)	0.887(0.13)	0.434(0.04)	0.582(0.07)	0.866(0.13)	0.43(0.04)	0.573(0.06)
PDE	0.774(0.18)	0.798(0.0)	0.774(0.1)	0.774(0.18)	0.448(0.05)	0.563(0.09)	0.774(0.18)	0.443(0.06)	0.56(0.09)
1-NN	1.0(0.0)	0.798(0.0)	0.888(0.0)	0.993(0.02)	0.42(0.01)	0.59(0.01)	1.0(0.0)	0.423(0.01)	**0.594(0.01)**
k-NN	0.96(0.01)	0.796(0.01)	0.87(0.01)	0.971(0.07)	0.438(0.02)	0.604(0.03)	0.936(0.08)	0.426(0.03)	0.585(0.04)
SVDD	0.625(0.02)	0.805(0.02)	0.704(0.02)	0.317(0.15)	0.467(0.15)	0.368(0.15)	0.282(0.13)	0.48(0.18)	0.343(0.15)

The bolded value is the best within the group.

Table 5.7: The average recall, precision and F_1-score of WBC data depending on the noise level (standard deviation in parentheses)

0%									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.947(0.01)	1.0(0.0)	0.973(0.0)	0.953(0.05)	0.948(0.06)	0.95(0.05)	0.93(0.07)	0.943(0.06)	0.934(0.05)
max norm	0.939(0.01)	1.0(0.0)	0.969(0.0)	0.93(0.05)	0.942(0.05)	0.935(0.04)	0.93(0.05)	0.942(0.05)	0.935(0.04)
one norm	0.948(0.02)	1.0(0.0)	0.973(0.01)	0.958(0.06)	0.956(0.04)	0.956(0.04)	0.944(0.07)	0.945(0.04)	**0.943(0.05)**
mix norm	0.921(0.02)	1.0(0.0)	0.959(0.01)	0.92(0.06)	0.942(0.05)	0.929(0.04)	0.93(0.05)	0.946(0.05)	0.937(0.04)
MoG	0.999(0.0)	1.0(0.0)	1.0(0.0)	0.902(0.07)	0.962(0.05)	0.929(0.05)	0.864(0.08)	0.97(0.04)	0.912(0.06)
PDE	0.92(0.07)	1.0(0.0)	0.957(0.04)	0.92(0.07)	0.962(0.04)	0.939(0.05)	0.92(0.07)	0.962(0.04)	**0.939(0.05)**
1-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.859(0.07)	0.567(0.07)	0.682(0.07)	0.854(0.07)	0.552(0.05)	0.67(0.05)
k-NN	0.941(0.01)	1.0(0.0)	0.97(0.01)	0.944(0.04)	0.904(0.05)	0.923(0.03)	0.939(0.06)	0.877(0.08)	0.905(0.05)
SVDD	0.959(0.01)	1.0(0.0)	0.979(0.0)	0.912(0.06)	0.922(0.06)	0.916(0.05)	0.921(0.05)	0.907(0.07)	0.913(0.06)
5 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.962(0.01)	0.986(0.0)	0.974(0.0)	0.953(0.05)	0.928(0.07)	0.939(0.05)	0.958(0.04)	0.941(0.05)	**0.949(0.04)**
max norm	0.939(0.01)	0.99(0.0)	0.964(0.0)	0.93(0.05)	0.937(0.06)	0.933(0.05)	0.93(0.05)	0.935(0.05)	0.932(0.05)
one norm	0.932(0.02)	0.996(0.0)	0.963(0.01)	0.916(0.07)	0.97(0.03)	0.941(0.05)	0.92(0.07)	0.97(0.02)	0.944(0.04)
mix norm	0.915(0.03)	0.99(0.0)	0.951(0.02)	0.902(0.07)	0.951(0.04)	0.925(0.05)	0.912(0.06)	0.953(0.05)	0.931(0.04)
MoG	0.996(0.01)	0.994(0.01)	0.995(0.0)	0.892(0.06)	0.967(0.04)	0.927(0.04)	0.878(0.09)	0.961(0.07)	0.914(0.07)
PDE	0.92(0.07)	0.947(0.0)	0.932(0.04)	0.92(0.07)	0.961(0.03)	0.939(0.05)	0.925(0.07)	0.961(0.03)	**0.942(0.05)**
1-NN	1.0(0.0)	0.947(0.0)	0.973(0.0)	0.995(0.01)	0.472(0.01)	0.641(0.01)	0.995(0.01)	0.472(0.01)	0.641(0.01)
k-NN	0.888(0.02)	0.988(0.01)	0.935(0.01)	0.92(0.06)	0.837(0.06)	0.876(0.05)	0.901(0.09)	0.833(0.08)	0.862(0.06)
SVDD	0.94(0.01)	0.967(0.01)	0.953(0.01)	0.884(0.06)	0.925(0.06)	0.902(0.05)	0.883(0.05)	0.919(0.05)	0.9(0.04)
10 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.92(0.05)	0.987(0.01)	0.951(0.02)	0.921(0.06)	0.967(0.04)	0.942(0.04)	0.935(0.05)	0.962(0.04)	**0.947(0.04)**
max norm	0.936(0.01)	0.978(0.01)	0.957(0.01)	0.93(0.05)	0.932(0.05)	0.93(0.04)	0.935(0.05)	0.933(0.05)	0.933(0.04)
one norm	0.936(0.01)	0.99(0.0)	0.962(0.01)	0.929(0.07)	0.965(0.03)	0.946(0.05)	0.929(0.07)	0.966(0.03)	0.946(0.04)
mix norm	0.923(0.01)	0.982(0.01)	0.952(0.01)	0.916(0.07)	0.937(0.06)	0.924(0.05)	0.916(0.06)	0.932(0.06)	0.923(0.05)
MoG	0.998(0.0)	0.999(0.0)	0.999(0.0)	0.882(0.09)	0.985(0.02)	0.928(0.05)	0.859(0.1)	0.971(0.03)	0.907(0.06)
PDE	0.925(0.07)	0.899(0.0)	0.91(0.04)	0.925(0.07)	0.961(0.04)	0.941(0.05)	0.93(0.07)	0.962(0.03)	**0.944(0.04)**
1-NN	1.0(0.0)	0.899(0.0)	0.947(0.0)	0.995(0.01)	0.473(0.01)	0.642(0.01)	0.995(0.01)	0.472(0.01)	0.641(0.01)
k-NN	0.826(0.03)	0.962(0.01)	0.889(0.02)	0.817(0.09)	0.724(0.09)	0.762(0.07)	0.807(0.1)	0.707(0.06)	0.749(0.05)
SVDD	0.944(0.01)	0.939(0.01)	0.941(0.01)	0.893(0.06)	0.92(0.05)	0.905(0.05)	0.879(0.05)	0.902(0.05)	0.889(0.04)

15%									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.965(0.01)	0.955(0.01)	0.96(0.0)	0.963(0.04)	0.917(0.06)	0.939(0.05)	0.963(0.04)	0.915(0.07)	0.937(0.05)
max norm	0.936(0.01)	0.967(0.0)	0.951(0.01)	0.93(0.05)	0.932(0.05)	0.93(0.04)	0.935(0.05)	0.933(0.05)	0.933(0.04)
one norm	0.937(0.01)	0.991(0.0)	0.963(0.01)	0.925(0.07)	0.974(0.03)	0.948(0.05)	0.925(0.07)	0.971(0.03)	**0.946(0.05)**
mix norm	0.932(0.01)	0.974(0.01)	0.953(0.01)	0.921(0.06)	0.94(0.05)	0.929(0.04)	0.935(0.04)	0.945(0.05)	0.939(0.04)
MoG	0.937(0.01)	0.966(0.01)	0.951(0.0)	0.921(0.05)	0.935(0.04)	0.927(0.04)	0.916(0.04)	0.926(0.05)	0.92(0.04)
PDE	0.939(0.06)	0.971(0.0)	0.954(0.03)	0.939(0.06)	0.944(0.04)	0.941(0.04)	0.939(0.06)	0.941(0.05)	**0.939(0.05)**
1-NN	1.0(0.0)	0.849(0.0)	0.918(0.0)	0.995(0.01)	0.472(0.01)	0.641(0.01)	0.995(0.01)	0.472(0.01)	0.641(0.01)
k-NN	0.763(0.02)	0.914(0.02)	0.832(0.02)	0.746(0.08)	0.618(0.03)	0.675(0.05)	0.727(0.12)	0.608(0.1)	0.659(0.1)
SVDD	0.904(0.01)	0.959(0.01)	0.93(0.01)	0.87(0.06)	0.917(0.03)	0.891(0.04)	0.86(0.07)	0.917(0.05)	0.886(0.04)
20 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.94(0.01)	0.972(0.01)	0.955(0.01)	0.944(0.05)	0.953(0.04)	0.948(0.04)	0.934(0.07)	0.948(0.04)	0.94(0.05)
max norm	0.936(0.01)	0.958(0.01)	0.947(0.01)	0.93(0.05)	0.937(0.05)	0.932(0.04)	0.93(0.05)	0.937(0.06)	0.932(0.04)
one norm	0.943(0.02)	0.975(0.02)	0.958(0.01)	0.944(0.07)	0.961(0.02)	0.951(0.04)	0.943(0.08)	0.949(0.03)	**0.945(0.04)**
mix norm	0.898(0.02)	0.975(0.01)	0.935(0.01)	0.92(0.06)	0.948(0.05)	0.933(0.05)	0.883(0.06)	0.952(0.04)	0.914(0.03)
MoG	0.996(0.0)	0.973(0.02)	0.984(0.01)	0.911(0.07)	0.958(0.04)	0.932(0.05)	0.878(0.09)	0.938(0.07)	0.902(0.05)
PDE	0.925(0.07)	0.798(0.0)	0.855(0.03)	0.925(0.07)	0.96(0.03)	0.941(0.05)	0.93(0.07)	0.961(0.03)	**0.944(0.05)**
1-NN	1.0(0.0)	0.798(0.0)	0.887(0.0)	0.995(0.01)	0.473(0.01)	0.642(0.01)	0.995(0.01)	0.472(0.01)	0.641(0.01)
k-NN	0.981(0.01)	0.801(0.01)	0.882(0.01)	1.0(0.0)	0.48(0.01)	0.649(0.01)	0.991(0.03)	0.476(0.01)	0.643(0.02)
SVDD	0.922(0.01)	0.919(0.01)	0.92(0.01)	0.888(0.06)	0.886(0.05)	0.886(0.04)	0.884(0.06)	0.871(0.06)	0.875(0.04)

The bolded value is the best within the group.

Table 5.8: The average recall, precision and F_1-score of Ionosphere data depending on the noise level (standard deviation in parentheses)

0%									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.991(0.0)	1.0(0.0)	0.996(0.0)	0.982(0.03)	0.908(0.07)	0.942(0.04)	0.973(0.03)	0.901(0.06)	0.935(0.04)
max norm	0.956(0.01)	1.0(0.0)	0.977(0.01)	0.934(0.04)	0.961(0.04)	0.947(0.03)	0.925(0.04)	0.969(0.03)	**0.946(0.03)**
one norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.978(0.02)	0.899(0.06)	0.935(0.03)	0.969(0.03)	0.869(0.06)	0.915(0.05)
mix norm	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.991(0.03)	0.77(0.09)	0.863(0.05)	0.996(0.01)	0.763(0.09)	0.861(0.05)
MoG	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.99(0.03)	0.93(0.06)	0.958(0.03)	0.96(0.05)	0.913(0.07)	0.933(0.04)
PDE	0.961(0.05)	1.0(0.0)	0.979(0.02)	0.961(0.05)	0.932(0.06)	0.944(0.03)	0.96(0.04)	0.932(0.06)	**0.944(0.03)**
1-NN	1.0(0.0)	1.0(0.0)	1.0(0.0)	0.938(0.04)	0.785(0.08)	0.853(0.06)	0.92(0.06)	0.764(0.06)	0.832(0.04)
k-NN	0.928(0.02)	1.0(0.0)	0.963(0.01)	0.9(0.13)	0.882(0.07)	0.883(0.08)	0.878(0.16)	0.843(0.07)	0.854(0.1)
SVDD	0.97(0.01)	1.0(0.0)	0.984(0.0)	0.921(0.08)	0.902(0.07)	0.908(0.05)	0.926(0.07)	0.899(0.07)	0.909(0.05)
5 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.963(0.01)	0.995(0.0)	0.979(0.01)	0.924(0.07)	0.937(0.05)	0.928(0.04)	0.929(0.06)	0.938(0.05)	0.931(0.03)
max norm	0.998(0.0)	0.984(0.0)	0.991(0.0)	0.965(0.03)	0.913(0.08)	0.935(0.04)	0.956(0.04)	0.927(0.08)	0.938(0.04)
one norm	1.0(0.0)	0.951(0.0)	0.975(0.0)	0.991(0.01)	0.968(0.02)	0.98(0.01)	0.991(0.02)	0.969(0.02)	0.98(0.01)
mix norm	1.0(0.0)	0.949(0.0)	0.974(0.0)	1.0(0.0)	0.643(0.01)	0.783(0.01)	1.0(0.0)	0.643(0.01)	**0.783(0.01)**
MoG	0.989(0.0)	0.996(0.0)	0.992(0.0)	0.899(0.08)	0.954(0.07)	0.922(0.05)	0.89(0.08)	0.95(0.07)	0.915(0.05)
PDE	0.961(0.05)	0.949(0.0)	0.954(0.02)	0.961(0.05)	0.93(0.08)	0.943(0.05)	0.96(0.04)	0.93(0.08)	**0.943(0.04)**
1-NN	1.0(0.0)	0.949(0.0)	0.974(0.0)	0.951(0.03)	0.773(0.07)	0.851(0.05)	0.942(0.05)	0.762(0.07)	0.839(0.03)
k-NN	0.9(0.02)	0.983(0.0)	0.939(0.01)	0.899(0.1)	0.864(0.06)	0.875(0.05)	0.872(0.1)	0.842(0.07)	0.853(0.06)
SVDD	0.949(0.01)	0.97(0.01)	0.96(0.01)	0.863(0.08)	0.937(0.07)	0.893(0.04)	0.863(0.06)	0.923(0.07)	0.888(0.04)
10 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.971(0.01)	0.987(0.0)	0.979(0.01)	0.956(0.04)	0.917(0.07)	0.934(0.04)	0.938(0.07)	0.904(0.08)	0.918(0.06)
max norm	1.0(0.0)	0.9(0.0)	0.947(0.0)	0.961(0.03)	0.907(0.08)	0.931(0.04)	0.969(0.03)	0.907(0.07)	**0.935(0.03)**
one norm	0.991(0.01)	0.98(0.0)	0.986(0.0)	0.951(0.05)	0.902(0.07)	0.923(0.04)	0.956(0.05)	0.876(0.07)	0.912(0.05)
mix norm	0.987(0.04)	0.904(0.01)	0.943(0.01)	0.991(0.03)	0.655(0.04)	0.787(0.02)	0.987(0.04)	0.651(0.02)	0.783(0.0)
MoG	0.991(0.0)	0.996(0.0)	0.993(0.0)	0.899(0.08)	0.956(0.06)	0.924(0.06)	0.899(0.08)	0.946(0.06)	0.92(0.06)
PDE	0.961(0.05)	0.9(0.0)	0.929(0.02)	0.961(0.05)	0.927(0.07)	0.942(0.05)	0.96(0.04)	0.928(0.07)	**0.942(0.04)**
1-NN	1.0(0.0)	0.9(0.0)	0.947(0.0)	0.951(0.03)	0.743(0.05)	0.834(0.04)	0.942(0.05)	0.741(0.04)	0.828(0.03)
k-NN	0.95(0.02)	0.962(0.0)	0.956(0.01)	0.939(0.08)	0.813(0.06)	0.87(0.06)	0.922(0.11)	0.812(0.05)	0.861(0.07)
SVDD	0.958(0.01)	0.94(0.01)	0.949(0.01)	0.876(0.08)	0.882(0.07)	0.876(0.05)	0.89(0.07)	0.877(0.07)	0.88(0.04)

15%									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.968(0.01)	0.98(0.01)	0.974(0.01)	0.956(0.05)	0.924(0.06)	0.938(0.04)	0.942(0.05)	0.907(0.1)	0.919(0.05)
max norm	0.977(0.01)	0.987(0.0)	0.982(0.0)	0.956(0.04)	0.925(0.07)	0.939(0.04)	0.938(0.04)	0.933(0.08)	**0.934(0.05)**
one norm	0.991(0.01)	0.951(0.01)	0.971(0.01)	0.991(0.02)	0.871(0.08)	0.925(0.05)	0.982(0.03)	0.852(0.09)	0.91(0.05)
mix norm	0.997(0.01)	0.858(0.03)	0.922(0.01)	1.0(0.0)	0.656(0.04)	0.792(0.03)	1.0(0.0)	0.651(0.02)	0.788(0.02)
MoG	0.989(0.0)	0.996(0.0)	0.992(0.0)	0.899(0.08)	0.948(0.06)	0.92(0.06)	0.89(0.08)	0.956(0.06)	0.919(0.06)
PDE	0.961(0.05)	0.849(0.0)	0.901(0.02)	0.961(0.05)	0.927(0.06)	0.943(0.05)	0.96(0.04)	0.927(0.06)	**0.943(0.05)**
1-NN	1.0(0.0)	0.849(0.0)	0.918(0.0)	0.96(0.03)	0.751(0.06)	0.841(0.04)	0.96(0.04)	0.724(0.06)	0.825(0.05)
k-NN	0.957(0.01)	0.925(0.01)	0.941(0.01)	0.934(0.05)	0.818(0.07)	0.872(0.06)	0.907(0.07)	0.782(0.06)	0.839(0.06)
SVDD	0.962(0.01)	0.904(0.01)	0.932(0.01)	0.894(0.08)	0.853(0.1)	0.869(0.07)	0.903(0.08)	0.853(0.1)	0.874(0.07)
20 %									
	Training			Validation			Test		
classifier	recall	precision	F_1-score	recall	precision	F_1-score	recall	precision	F_1-score
two norm	0.962(0.01)	0.967(0.01)	0.964(0.01)	0.933(0.05)	0.928(0.05)	0.929(0.04)	0.942(0.06)	0.924(0.04)	0.932(0.04)
max norm	0.994(0.0)	0.959(0.01)	0.976(0.01)	0.97(0.03)	0.927(0.05)	0.947(0.04)	0.956(0.04)	0.918(0.06)	**0.936(0.04)**
one norm	0.988(0.01)	0.936(0.01)	0.961(0.0)	0.974(0.03)	0.871(0.06)	0.918(0.04)	0.947(0.07)	0.862(0.06)	0.899(0.04)
mix norm	0.984(0.04)	0.817(0.04)	0.891(0.01)	0.996(0.01)	0.671(0.05)	0.8(0.03)	1.0(0.0)	0.675(0.07)	0.804(0.04)
MoG	0.988(0.0)	0.995(0.0)	0.991(0.0)	0.899(0.08)	0.944(0.06)	0.918(0.06)	0.886(0.08)	0.943(0.06)	0.911(0.06)
PDE	0.961(0.05)	0.798(0.0)	0.871(0.02)	0.961(0.05)	0.927(0.05)	0.942(0.03)	0.96(0.04)	0.927(0.05)	**0.942(0.03)**
1-NN	1.0(0.0)	0.798(0.0)	0.888(0.0)	0.987(0.02)	0.689(0.03)	0.811(0.02)	0.973(0.03)	0.686(0.02)	0.804(0.01)
k-NN	0.969(0.01)	0.868(0.02)	0.916(0.01)	0.965(0.04)	0.771(0.03)	0.856(0.02)	0.93(0.1)	0.738(0.06)	0.821(0.07)
SVDD	0.967(0.01)	0.854(0.01)	0.907(0.01)	0.912(0.07)	0.842(0.05)	0.875(0.05)	0.916(0.06)	0.84(0.05)	0.875(0.05)

The bolded value is the best within the group.

Bibliography

[1] M. T. Beck, M. Werner, S. Feld, and T. Schimper, *Mobile edge computing: A taxonomy*, 2014.

[2] C. M. Bishop, *Novelty detection and neural network validation*, IEE Proceedings-Vision, Image and Signal processing, vol. 141, 1994, pp. 217–222.

[3] C. M. Bishop, *Neural networks for pattern recognition*, Oxford university press, 1995.

[4] G. G. Cabral, A. L. Oliveira, and C. B. Cahú, *Combining nearest neighbor data description and structural risk minimization for one-class classification*, Neural Computing and Applications, vol. 18, 2009, pp. 175–183.

[5] M. Conforti, G. Cornuéjols, and G. Zambelli, *Integer programming*, vol. 271, Springer, 2014.

[6] C. Cortes and V. Vapnik, *Support-vector networks*, Machine learning, vol. 20, 1995, pp. 273–297.

[7] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern classification*, John Wiley & Sons, 2012.

[8] J. Eckstein, P. L. Hammer, Y. Liu, M. Nediak, and B. Simeone, *The maximum box problem and its application to data analysis*, Computational Optimization and Applications, vol. 23, 2002, pp. 285–298.

[9] M. M. Gaber, J. B. Gomes, and F. Stahl, *Pocket data mining*, Big Data on Small Devices. Series: Studies in Big Data, 2014.

[10] T. Hastie, R. Tibshirani, and M. Wainwright, *Statistical learning with sparsity*, CRC press, 2015.

[11] X. Hong, S. Chen, and V. M. Becerra, *Sparse density estimator with tunable kernels*, Neurocomputing, vol. 173, 2016, pp. 1976–1982.

[12] N. Japkowicz, C. Myers, M. Gluck, et al., *A novelty detection approach to classification*, in IJCAI, vol. 1, 1995, pp. 518–523.

[13] I. K. Jeong and J. Y. Choi, *Design of one-class classifier using hyper-rectangles*, Journal of Korean Institute of Industrial Engineers, vol. 41, 2015, pp. 439–446.

[14] F. Letouzey, F. Denis, and R. Gilleron, *Learning from positive and unlabeled examples*, in Algorithmic Learning Theory, Springer, 2000, pp. 71–85.

[15] C. Li, Y. Zhang, and X. Li, *Ocvfdt: one-class very fast decision tree for one-class classification of data streams*, in Proceedings of the Third International Workshop on Knowledge Discovery from Sensor Data, ACM, 2009, pp. 79–86.

[16] M. Lichman, UCI machine learning repository, http://archive.ics.uci.edu/ml, 2013.

[17] L. Manevitz and M. Yousef, *One-class document classification via neural networks*, Neurocomputing, vol. 70, 2007, pp. 1466–1481.

[18] M. M. Moya, M. W. Koch, and L. D. Hostetler, *One-class classifier networks for target recognition applications*, tech. rep., Sandia National Labs., Albuquerque, NM (United States), 1993.

[19] E. Parzen, *On estimation of a probability density function and mode*, The annals of mathematical statistics, vol. 33, 1962, pp. 1065–1076.

[20] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, *Scikit-learn: Machine learning in Python*, Journal of Machine Learning Research, vol. 12, 2011, pp. 2825–2830.

[21] S. L. Phung, A. Bouzerdoum, and G. H. Nguyen, *Learning pattern classification tasks with imbalanced data sets*, 2009.

[22] G. Ritter and M. T. Gallegos, *Outliers in statistical pattern recognition and an application to automatic chromosome classification*, Pattern Recognition Letters, vol. 18, 1997, pp. 525–539.

[23] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, J. C. Platt, et al., *Support vector method for novelty detection.*, in NIPS, vol. 12, 1999, pp. 582–588.

[24] P. Serafini, *Classifying negative and positive points by optimal box clustering*, Discrete Applied Mathematics, vol. 165, 2014, pp. 270–282.

[25] A. Skabar, *Single-class classifier learning using neural networks: An application to the prediction of mineral deposits*, in Machine Learning and Cybernetics, 2003 International Conference on, vol. 4, IEEE, 2003, pp. 2127–2132.

[26] D. M. Tax and R. P. Duin, *Data domain description using support vectors.*, in ESANN, vol. 99, 1999, pp. 251–256.

[27] D. M. Tax and R. P. Duin, *Data description in subspaces*, in Pattern Recognition, 2000. Proceedings. 15th International Conference on, vol. 2, IEEE, 2000, pp. 672–675.

[28] D. M. Tax and R. P. Duin, *Support vector data description*, Machine learning, vol. 54, 2004, pp. 45–66.

[29] D. M. J. Tax, *One-class classification*, PhD thesis, Delft University of Technology, 2001.

[30] N. F. Thornhill, S. C. Patwardhan, and S. L. Shah, The continuous stirred tank heater simulation, http://personal-pages.ps.ic.ac.uk/ nina/CSTHSimulation/index.htm, 2013.

[31] N. F. Thornhill, S. C. Patwardhan, and S. L. Shah, *A continuous stirred tank heater simulation model with applications*, Journal of Process Control, vol. 18, 2008, pp. 347–360.

[32] C. F. Tsai, Y. F. Hsu, C. Y. Lin, and W. Y. Lin, *Intrusion detection by machine learning: A review*, Expert Systems with Applications, vol. 36, 2009, pp. 11994–12000.

[33] Xpress, Xpress 8.0, http://www.fico.com/en, 2017.

[34] H. Yu, J. Han, and K. C. Chang, *Pebl: Web page classification without negative examples*, IEEE Transactions on Knowledge and Data Engineering, vol. 16, 2004, pp. 70–81.

[35] H. Yu, C. Zhai, and J. Han, *Text classification from positive and unlabeled documents*, in Proceedings of the twelfth international conference on Information and knowledge management, ACM, 2003, pp. 232–239.

[36] J. Zhou, K. Chan, V. Chong, and S. M. Krishnan, *Extraction of brain tumor from mr images using one-class support vector machine*, in Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the, IEEE, 2006, pp. 6411–6414.

국문초록

본 논문에서는 노름 공(norm ball)을 이용한 새로운 단일 클래스 분류기를 제안한다. 주어진 타겟 데이터를 덮을 수 있는 유한개의 노름 공들이 존재할 때, 이들을 이용해 타겟 데이터만을 분류하는 노름 공 분류기를 구축하는 것이 목적이다. 분류기를 구축하기에 앞서 유한 개의 노름 공을 생성해야 한다. 논문의 알고리즘을 적용시키면, 주어진 타겟 데이터를 이용해 최대 데이터 갯수의 제곱에 해당하는 일차적인 노름 공 후보를 생성할 수 있다. 좀 더 타겟 데이터를 잘 설명하기 위해 2가지 조건을 적용시켜 데이터를 설명하기에 더 적합한 최종 노름 공 후보를 선정한다. 선정한 최종 후보들로 0-1 정수 계획 모델을 세우고 타겟 데이터를 잘 설명하는 최적 노름 공들 찾는다. 이 때, 목적함수는 에러와 선택되는 노름 공의 갯수를 최소화 하는 것이다. 최적 노름 공들로 단일 클래스 분류기를 구축하게 된다. 제안한 단일 분류기의 평가를 위해 다른 단일 클래스 분류기들을 선정해 비교해 보았다. 실험 결과 제안된 단일 클래스 분류기는 비교군과 비교해 보았을 때 성능이 나쁘지 않음을 확인할 수 있었다. 또한, 다른 분류기들에 비해 모델의 복잡성이 낮았고, 그로 인해 새로운 관측치들을 분류하는 것에 매우 짧은 시간이 걸렸다. 잡음 존재 시 분류기 성능 평가 실험에서는 최대 노름(maximum norm)을 이용해 구축한 단일 분류기가 다른 종류의 노름을 사용했을 때보다 잡음에 덜 민감하고 안정적인 성능을 보여준다는 것을 알 수 있었다.

주요어: 단일 분류, 노름 공 덮개, 정수 계획 모형

학번: 2015-22855